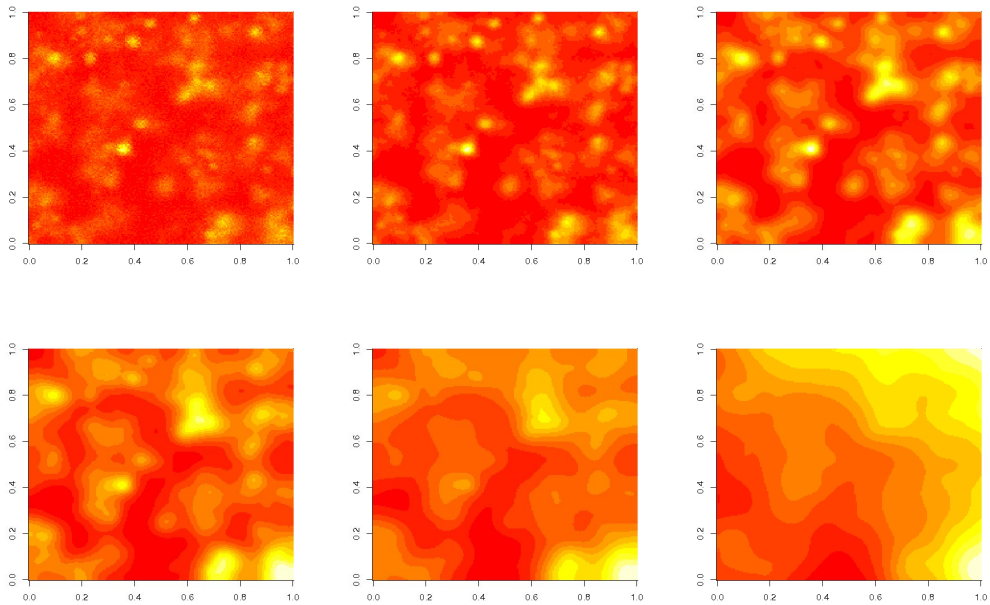
Signal and Image Denoising Using Inhomogeneous Diffusion



Dissertation

zur Erlangung des akademischen Grades
Doktor der Naturwissenschaften
(Dr. rer. nat.)
dem Fachbereich Mathematik der Universität Duisburg-Essen vorgelegt von
Rahel Stichtenoth aus Essen

Dezember 2007

Tag der Disputation: 08.02.2008

Gutachter: Prof. Dr. P.L. Davies (Universität Duisburg-Essen)
Prof. Dr. A. Munk (Georg-August-Universität Göttingen)

The sequence of images on the titlepage illustrates the contents of this thesis. The first image shows the Quantum Computing dataset (cf. Appendix A.2.3) interpreted as a heat distribution, and the successive images show the homogeneous diffusion process stopped at different times $\tau = 10$, 50, 250, 2000 and 10^5. None of these images provides a satisfactory approximation of the data. A much better approximation can be achieved by using the inhomogeneous diffusion estimator $\hat{f}_a$, which is developed in this thesis. The result is shown in Section 3.6 on Page 75.

Abstract

A huge amount of data needs to be processed these days. In many fields one wishes to interpret given datasets, which are often corrupted by noise. The development of efficient methods of denoising therefore is a challenging area of research. The need also arises in connection with many applications, e.g. signal processing in measurement and control technique, medical image analysis, spectroscopy and sensors in digital cameras.
This thesis is concerned with a new denoising method. We use a nonparametric approach where no prior information on the distribution of the data is assumed, and essentially focus on smooth datasets.
In the first chapter we describe some nonparametric regression methods and discuss the problems concerning the selection of the smoothing parameters. In case of datasets with varying smoothness, estimators with a local smoothing parameter are preferred, naturally, to those with one global smoothing parameter. The smoothing parameter of the Nadaraya-Watson kernel estimator for instance can be localized. However it is not suitable for denoising two-dimensional datasets since it takes relatively long to compute it. Similar drawbacks of other known methods are pointed out in Chapter 1, to show that our method can be utilized with advantage. Indeed, not only the computing time is reduced considerably by the use of our method, but also smoother results can be obtained.
We introduce the novel diffusion estimator $\hat{f}_\tau$ and its localized version $\hat{f}_a$ in the second chapter for the one-dimensional setting. We give a brief description of the finite differences method, which we use to compute $\hat{f}_\tau$ and $\hat{f}_a$ by solving particular differential equations numerically. The local smoothing parameter is selected with an iterative algorithm using the so called multiresolution criterion. In each iteration step, a statistical analysis of the residuals is made. The smoothing parameter is adapted such that eventually the residuals contain only the noise, which is to be removed. A balance between the smoothness of the solution and the closeness to the data has to be achieved. It is due to this iterative algorithm that the computational speed is significant. A numerical comparison of our method to other nonparametric regression methods is also presented at the end of Chapter 2.
The third chapter is of main interest. It deals with the two-dimensional denoising problem. As the ingredients of our algorithm – the inhomogeneous diffusion process, its numerical solution and the choice of the smoothing parameter – are described in detail in the previous chapter, here the explanation is brief. In the two-dimensional setting, we additionally need a partition to be combined with the multiresolution criterion. This partition is also required to ensure reasonable computing time. Here we present two possible partitions, namely the partition into dyadic squares and the wedge partition. The results are compared, also to other

two-dimensional smoothing methods.
The rest of the work is of more theoretical nature. In the fourth chapter we show that the diffusion estimator $\hat{f}_\tau$ achieves the optimal rate of convergence. Chapter 5 provides the theoretical background for the multiresolution criterion. It deals with the modulus of continuity for the Brownian motion and the Brownian sheet. As Gaussian white noise can be embedded into the Brownian motion, respectively into the Brownian sheet, the modulus of continuity justifies the multiresolution criterion.
We close in Chapter 6 with a brief outlook on further research ideas linked to the work, presented here.
Appendix A gives a collection of all the datasets that have been used in this thesis.
The implementation of the diffusion estimator is realized in C and the statistical software R [Tea05]. The source code for an exemplary version of the diffusion estimator $\hat{f}_a$ can be found in Appendix B. The whole source code including different variations can be found on the webpage http://www.stat-math.uni-essen.de/∼stichtenoth.

Acknowledgements

First of all, I would like to thank my supervisor Laurie Davies for introducing me to the interesting field of image processing and to the fascination of applied mathematics in general. I very much appreciate his excellent supervision and encouragement throughout my PhD studies.
I would also like to thank the second supervisor Axel Munk for his willingness to review this thesis. I appreciate his careful reading and his useful comments.

The working atmosphere in our group has always been very enjoyable, in particular I am grateful to my colleagues Monika Meise, Christian Höhenrieder, Evgeny Zoldin and Jan Kalina. I always felt that I was welcome to ask questions whenever I needed to. Special thanks goes to Monika Meise for her outstanding commitment. Here I would like to mention that the implementation for the wedge partition is kindly provided by her.

A number of people from other universities and research centres have also contributed to this work. Among all the colleagues I have met during conferences, workshops and talks, I would like to point out Felix Friedrich for his help concerning the wedges and Harald Köstler, Joachim Weickert, Michael Breuß and Jörg Frochte, for their help related to the numerical solutions of PDEs.

It is needless to say that I also thank my family and friends for their support and help during my PhD studies. I will not mention all my personal thanks here, but outsource them to the personal copies of this work.

Last but not the least, I appreciate the financial support of the Colloborative Research Centre "Reduction of Complexity in Multivariate Data Structures" (SFB 475) of the German Research Foundation (DFG), in particular for making it possible for me to attend various international conferences and workshops which have always been very stimulating.

Contents

Chapter 1

Introduction

In this chapter, we give an introduction to nonparametric regression in one dimension. Various methods are presented in Section 1.2. They are based on the model introduced in Section 1.1. The aim of these methods is to find an adequate representation of a given dataset.
Section 1.3 intends to clarify the problems of parameter choice involved in the above methods. These problems lead to the development of a novel estimator presented in Chapter 2.

1.1 The Model

The basis for nonparametric regression methods is the following model:

$$Y_i = f(x_i) + \sigma\, Z_i, \quad Z_i \text{ i.i.d. } \mathcal{N}(0,1),$$

where σ is the standard deviation of the noise. Note that the noise here is restricted to be Gaussian white noise. In Section 6.1 we give some ideas for a generalization of this model.
For a given dataset

$$(x_i, y_i),\ i = 0, \ldots, n \quad \text{with } 0 = x_0 < x_1 \cdots < x_n = 1 \text{ and } y_i \in \mathbb{R}$$

we aim for a reasonable representation $\hat{f}_n(x_i)$, so that each y_i is composed of the value of $\hat{f}_n$ at x_i and some additive noise. In other words we wish to decompose the given data as

$$y_i = \hat{f}_n(x_i) + r_n(x_i),$$

where $\hat{f}_n$ is an estimator for the data y_i, depending on the sample size n, and the resulting residuals, $r_n(x_i)$, are expected to represent the noise.

See Figure 1.1, for example, where the Sine dataset (cf. Appendix A.1.1) is decomposed into the sine curve $f(x) = \sin(2\pi x)$ and the noise $0.1 \cdot Z_i$.

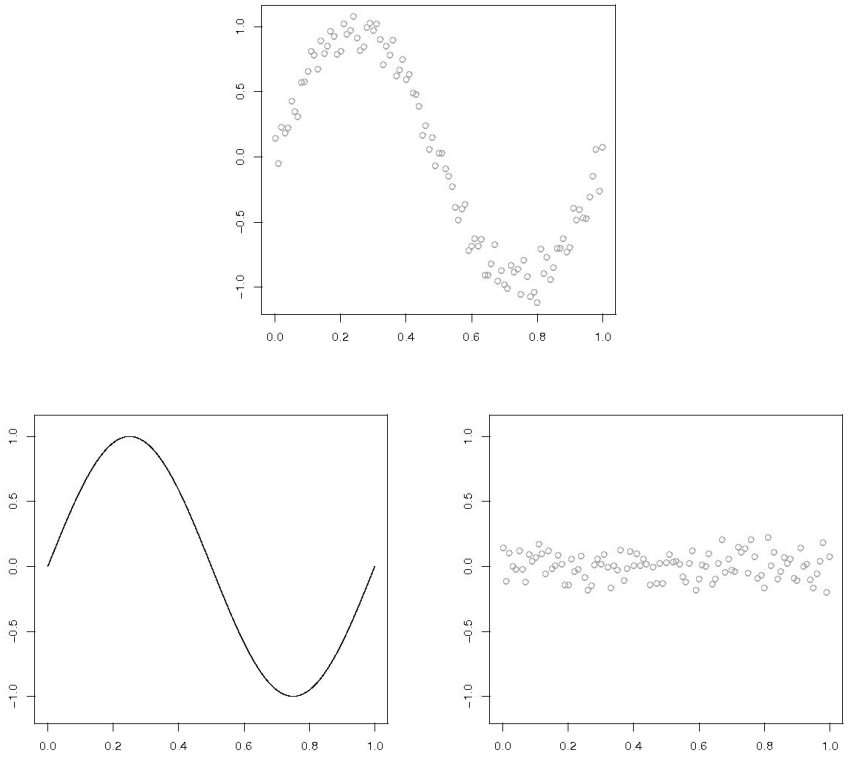


Figure 1.1: Sine dataset and perfect decomposition of the data into the values $f(x_i)$ of the sine function and the noise

The decomposition in this case is easy since the underlying function $f(x) = \sin(2\pi x)$ is known, whereas in practice the function f is unknown. Hence a smooth representation of the noisy data by the estimator $\hat{f}_n$ is sought for, so that the residuals $r_n(x_i) = y_i - \hat{f}_n(x_i)$ contain very little of the signal, but mainly the noise. Therefore it would be satisfactory to find an approximation of the sine function by an estimator $\hat{f}_n$ such that the residuals look like Gaussian white noise.

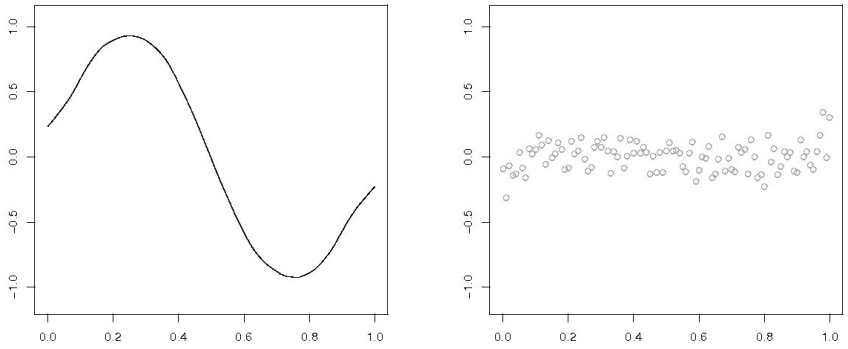


Figure 1.2: Decomposition of the Sine dataset into an estimator $\hat{f}_n(x_i)$ and the residuals $r_n(x_i)$. Here $\hat{f}_n$ is a kernel estimator with bandwidth $h = 0.1$.

1.2 Existing Nonparametric Regression Methods

The most popular nonparametric regression methods are kernel estimators, smoothing splines and wavelet shrinkage. We also mention the total variation regularization, the taut string and the adaptive weights smoothing methods, which have attracted considerable attention.
A brief overview of these methods is given below. An emphasis is put on kernel estimators, especially when the choice of the smoothing parameter is considered.

1.2.1 The Kernel Estimator

The kernel estimator has first been proposed by Nadaraya and Watson in 1964 (cf. Nadaraya [Nad64] and Watson [Wat64]).
For a kernel function $K(x)$ of order k, which satisfies

- $K(x) \geq 0$,
- $\int_{-\infty}^{\infty} K(x)\,dx = 1$,
- $\int_{-\infty}^{\infty} x^j K(x)\,dx = 0$ for $j = 1, \ldots, k-1$, $\int_{-\infty}^{\infty} x^k K(x)\,dx = \alpha \neq 0$,

the Nadaraya-Watson kernel estimator $f_{n,h}$ with bandwidth h is defined as

$$f_{n,h}(x) = \frac{\sum_{i=0}^{n} y_i \cdot K\left(\frac{x-x_i}{h}\right)}{\sum_{i=0}^{n} K\left(\frac{x-x_i}{h}\right)} .$$

Here h is the smoothing parameter.

1.2.2 Smoothing Splines

A smoothing spline is obtained as follows. Starting with the least squares estimator, one can add a roughness penalty in order to get a smoother solution. If one takes $||g''||_2^2$ as a measure for the roughness of an estimator, one is lead to the following minimization problem

$$\underset{g \in C^2([0,1])}{\operatorname{argmin}} S_\lambda(g) = \underset{g \in C^2([0,1])}{\operatorname{argmin}} \sum_{i=0}^{n} (y_i - g(x_i))^2 + \lambda ||g''||_2^2,$$

where λ is the smoothing parameter, which controls the trade-off between the data fidelity and the smoothness.
The unique solution among all twice differentiable functions is a cubic spline. In nonparametric regression it is called smoothing spline. It consists of cubic polynomials between the knots x_i with

- $p_i(x_i) = p_{i-1}(x_i)$,
- $p_i'(x_i) = p_{i-1}'(x_i)$,
- $p_i''(x_i) = p_{i-1}''(x_i)$.

This method provides a differentiable estimator. It is based on the L_2 norm. We refer to Härdle [Här90], Wahba [Wah90] and Green and Silverman [GS94] for a detailed explanation.

1.2.3 Total Variation Regularization

Rudin, Osher and Fatemi [ROF92] have proposed a similar method, namely the total variation regularization. They use the total variation $TV(f) = \int |f'(x)|\, dx$ as a roughness penalty. The data fidelity term is the squared L_2 norm, the same as for the smoothing spline. This results in the following minimization problem

$$\underset{g \in BV([0,1])}{\operatorname{argmin}} \left(\alpha ||y - g||_2^2 + TV(g)\right)$$

$$= \underset{g \in BV([0,1])}{\operatorname{argmin}} \left(\alpha \sum_{i=0}^{n} (y_i - g(x_i))^2 + \sum_{i=1}^{n} \sqrt{(g(x_i) - g(x_{i-1}))^2} \right),$$

where $BV([0,1])$ is the space of all functions on $[0,1]$ with bounded total variation, and α is the smoothing parameter.
The TV norm, which is essentially the L_1 norm of the derivative, is better suitable for images than the L_2 norm used for smoothing splines (cf. Rudin [Rud87]).

1.2.4 Wavelet Shrinkage

Another smoothing technique is the wavelet shrinkage, which was first proposed by Donoho and Johnstone [DJ94]. A wavelet transformation is a specific orthogonal series transformation. The dataset is transformed into the wavelet domain, and the coefficients of the wavelet series are analysed. Those coefficients, which do not pass a certain threshold, are set to be equal to zero, and the resulting wavelet series is then transformed back.
The most popular shrinkage techniques are the soft and the hard wavelet thresholding, given by the shrinkage functions

$$\delta_\lambda^H(x) = x \, \mathbb{1}_{\{|x|>\lambda\}} \quad \text{and}$$

$$\delta_\lambda^S(x) = \operatorname{sgn}(x) \, (|x| - \lambda)_+.$$

The smoothing parameter λ is contained, in this case, in the shrinkage function.
This method yields a simpler representation of the image, which can additionally be used for compression. The JPEG2000 standard is based on wavelet transformation.

1.2.5 The Taut String Method

For the taut string method, the data y_i are first integrated in order to obtain their empirical distribution function F_n. A tube of width ε is laid around F_n, and a string is taut within the tube. The derivative of the taut string is taken as an estimator for the data y_i. Among all the functions whose integral lies within the tube, it is the one with the minimal number of extreme values. The tube width ε is the smoothing parameter.
Davies and Kovac [DK01] have developed the idea of local squeezing: the tube-width becomes locally different, and the values for ε_i are obtained in an automatic and data driven procedure. The resulting estimator is the minimizer of the number of extreme values among all the functions satisfying the so called multiresolution criterion.

1.2.6 Adaptive Weights Smoothing

The adaptive weights smoothing method, due to Polzehl and Spokoiny [PS00], provides a locally constant estimator $\hat{f_{k^\star}}$, which is gained in an iterative procedure. The regions of constancy are chosen automatically and data adaptively. The value of $\hat{f_{k^\star}}$ in a region of constancy is obtained by a weighted mean value of the data within that region.

1.3 The Choice of the Smoothing Parameter

All the methods mentioned above include some smoothing parameter. Accordingly the problem of choosing the best smoothing parameter arises. We mention some strategies for the choice of the smoothing parameter below. We focus only on those for the kernel estimator since the estimator $\hat{f}_\tau$, which is introduced in this thesis, is based on the kernel estimator.
The asymptotically optimal bandwidth h_n is of order $n^{-1/5}$, which leads to the optimal rate of convergence. However it is of no use for finite data sets, which are encountered in reality.
For noisy Sine data, the asymptotically optimal bandwidth $h = n^{-1/5}$ yields reasonable results for sample sizes as of $n = 10^4$ (cf. Figure 1.3), but if the smoothness of the data varies like for the Sinepeak dataset, even for sample size $n = 10^5$ the results are far from being satisfactory (cf. Figure 1.4).

For finite datasets, the best bandwidth in terms of the least squares error can be chosen by cross-validation (Härdle and Marron [HM85]). But for datasets with local differences in smoothness, one global smoothing parameter for the whole dataset is simply not sufficient to ensure a smooth approximation.
The X-ray Diffractogram dataset clarifies the problem of a single global smoothing parameter (cf. Figure 1.5). This dataset is made available to us by courtesy of Dieter Mergel from the Physics Department of the University of Duisburg-Essen. X-ray Diffractograms are used to analyse the morphology of thin films. Important information is contained in the location, the power and the width of the peaks. Further information about this dataset is given in Appendix A.1.3.
In the computations we use the Epanechnikov kernel

$$K(x) = 0.75\ (1 - x)^2\ \mathbb{1}_{\{|x|<1\}}.$$

The computation of a kernel estimator using the Epanechnikov kernel is much faster than using the Gaussian kernel, as many of the summands in the case of the Epanechnikov kernel turn out to be zero. Therefore it is often preferred. One should note though that the Gaussian kernel is smooth itself and yields smoother

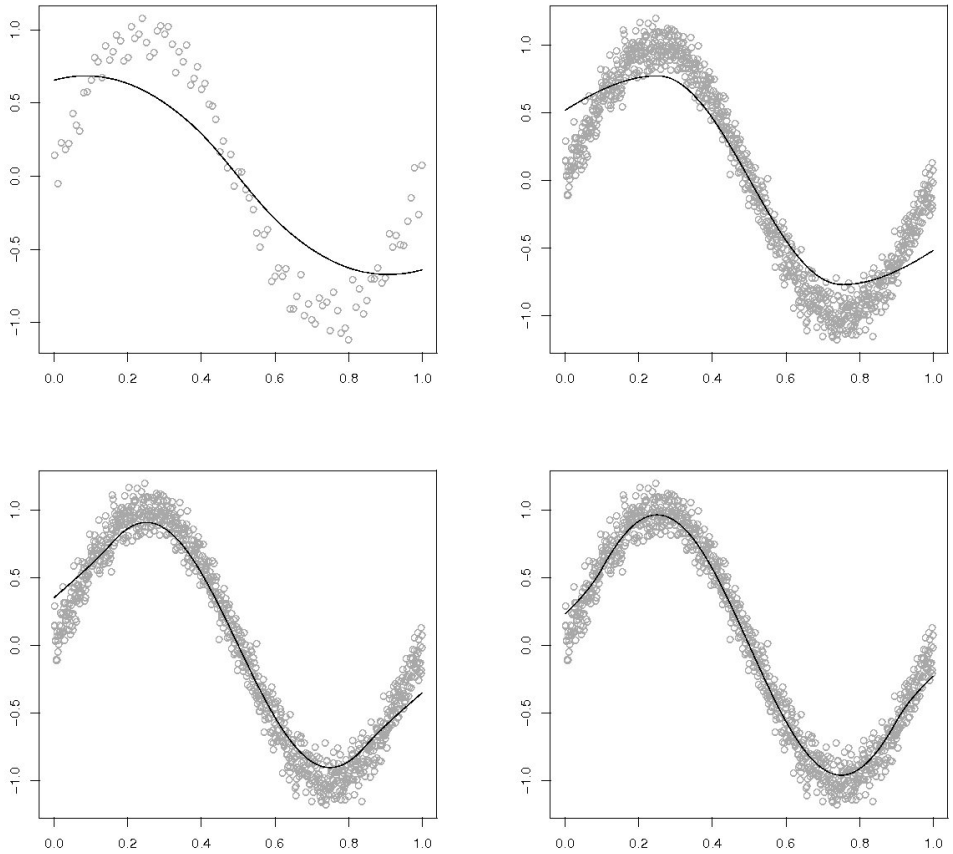


Figure 1.3: Kernel estimators $f_{n,h}$ with bandwidth $h = n^{-1/5}$ for the Sine dataset of different sample sizes $n = 100,\ n = 1000,\ n = 10^4$ and $n = 10^5$

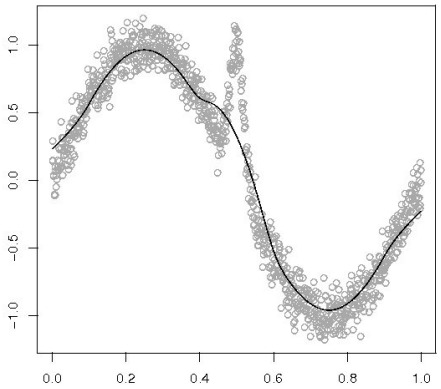


Figure 1.4: Kernel estimator $f_{n,h}$ with bandwidth $h = n^{-1/5}$ for the Sinepeak dataset of sample size $n = 10^5$

results whereas the Epanechnikov kernel with its discontinuities causes more artefacts.
A small bandwidth h approximates the peaks of the data but gives a rough approximation of the baseline. On the other hand a large bandwidth gives a smooth approximation of the baseline but looses a lot of information at the peaks, which are important features of the dataset. Hence both choices are not satisfactory.

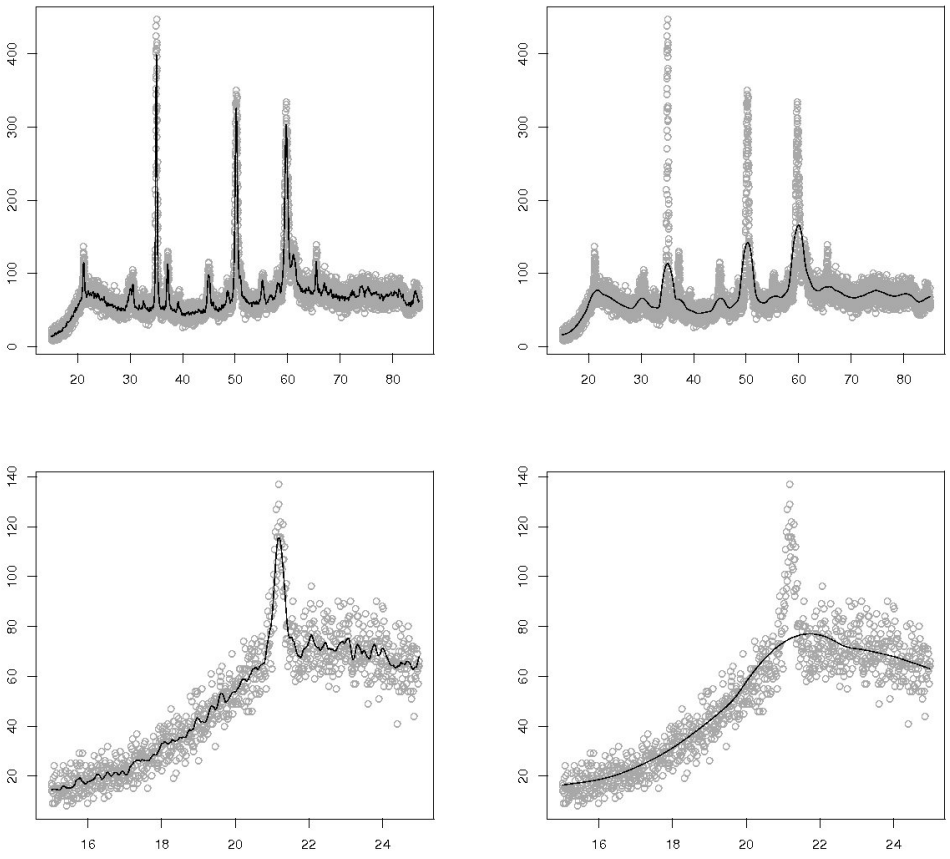


Figure 1.5: Kernel estimator $f_{n,h}$ with global bandwidths $h = 0.1$ and $h = 1.5$ for the X-ray Diffractogram dataset
Bottom: extract of the first 1000 data points

This problem can be overcome by localizing the bandwidth h, which results in the estimator

$$f_{n,h}(x) = \frac{\sum_{i=0}^{n} y_i \cdot K\left(\frac{x-x_i}{h(x_i)}\right)}{\sum_{i=0}^{n} K\left(\frac{x-x_i}{h(x_i)}\right)}$$

with the local smoothing parameter $h(x_i)$.
There are various strategies to select the local bandwidth $h(x_i)$, for instance, by cross-validation or plug-in methods. Figure 1.6 displays the result for the X-ray Diffractogram dataset using the plug-in bandwidth by Hermann [Her97].

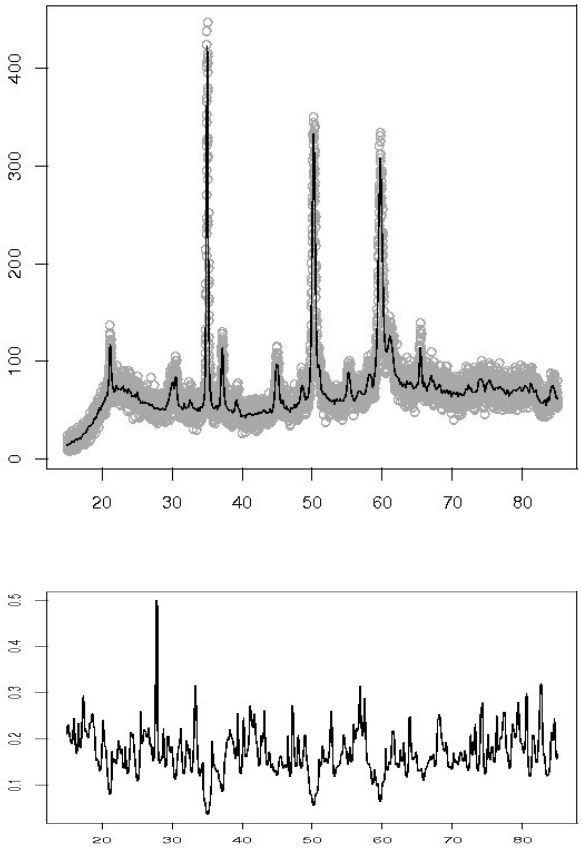


Figure 1.6: Kernel estimator with plug-in bandwidth for the X-ray Diffractogram dataset and its selected local bandwidth

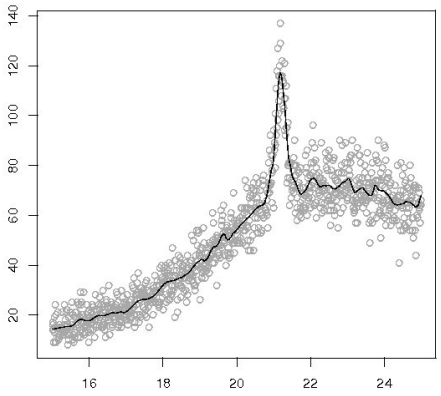


Figure 1.7: Extract of the first 1000 data points

The method is very fast (0.2 seconds) and approximates the peaks very well, however the smoothness of the estimator is still not satisfactory. This is clear in Figure 1.7, which shows an extract of the first 1000 data points.
Meise [Mei04] gives a bandwidth selection method using the multiresolution criterion of Davies and Kovac [DK01]. It is an iterative method and gives a smooth approximation, which at the same time stays close to the data. This is achieved by analysing the residuals on different scales for deciding whether they include only the noise or also parts of the signal. A more detailed explanation of the multiresolution criterion is given in Section 2.6.
The bandwidth $h(x_i)$ is selected automatically, and the selection is data driven. $h(x_i)$ is piecewise constant. Figure 1.8 shows the selected local bandwidth h and the resulting kernel estimator $f_{n,h}$.
Artefacts like the ones, that can be seen around 21° and 22° may appear where the value of the bandwidth jumps. This can be avoided if the bandwidth is additionally smoothed: First of all we search for jumps $x_{i_1}, \ldots, x_{i_k}$ with $h(x_{i_j}) \neq h(x_{i_j+1})$. Then we define ε_{i_j}-neighbourhoods for each jump x_{i_j}, where the values for ε_{i_j} are selected to be as large as possible so that the resulting estimator fulfills the multiresolution criterion. Within the ε_{i_j}-neighbourhoods, the bandwidth h is replaced by the smoothed bandwidth h_{smooth}, a polynomial of degree 3 with $h_{smooth}(x_{i_j} - \varepsilon_{i_j}) = h(x_{i_j} - \varepsilon_{i_j})$ and $h_{smooth}(x_{i_j} + \varepsilon_{i_j}) = h(x_{i_j} + \varepsilon_{i_j})$.
Figure 1.9 shows the smoothed bandwidth h_{smooth} and the resulting kernel estimator $f_{n,h_{smooth}}$.

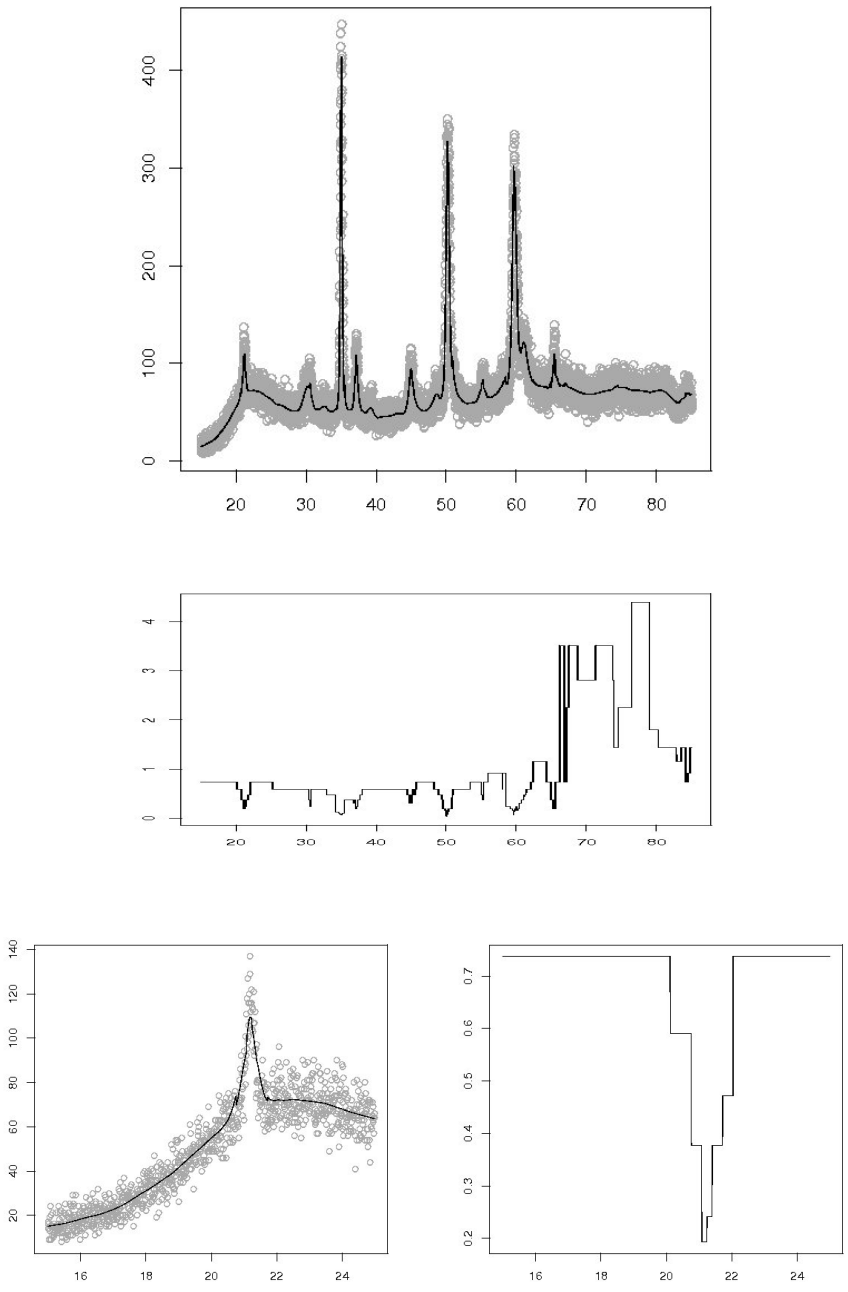


Figure 1.8: Kernel estimator $f_{n,h}$ and automatically selected local bandwidth $h(x_i)$ for the X-ray Diffractogram dataset
Bottom: extract of the first 1000 data points

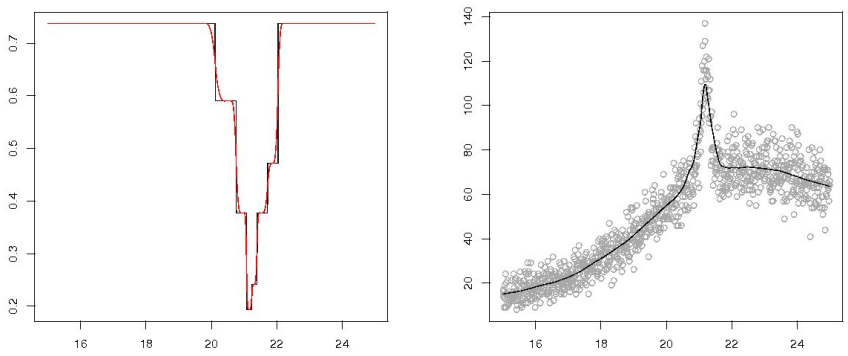


Figure 1.9: h (black) and h_{smooth} (red) and the kernel estimator $f_{n,h_{smooth}}$

This bandwidth selection method ensures that in regions of high variability in the data, h is chosen to be small in order to stay close to the data, whereas in smooth regions a large bandwidth is chosen to generate a smooth approximation. This can be seen in Figure 1.8, whereas for the plug-in bandwidth by Hermann, the connection between the selected bandwidth and the data is not obvious.
Unfortunately the bandwidth selection of Meise is rather slow. For the X-ray Diffractogram dataset it needs 117 seconds for 36 iteration steps. The computing time is due to the iterative procedure, where in each iteration step a kernel estimator with local bandwidth has to be computed. There are fast algorithms to compute the kernel estimator for a constant bandwidth h (cf. e.g. in Brockmann et al. [BEGS94]), but not for local bandwidths $h(x_i)$.
In this thesis we show that improvement of this method is possible by interpreting the kernel estimator as the solution of a partial differential equation (PDE), whose numerical solution can be computed very fast.

Chapter 2

The One-dimensional Diffusion Estimator

In this chapter we introduce the diffusion estimator $\hat{f}_\tau$ and its localized version $\hat{f}_a$. In order to obtain a reasonable approximation for a given dataset, even when its smoothness varies locally, we need the local smoothing parameter a. It will be chosen by the multiresolution criterion, which is presented in Section 2.6.
We start with the fact that $\hat{f}_\tau$ can be obtained by a transformation of the kernel estimator.

2.1 From the Kernel Estimator to the Diffusion Process

The kernel estimator with Gaussian kernel $K(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}x^2\right)$ and equidistant design points $x_i = \frac{i}{n}$ is defined as

$$f_{n,h}(x) = \frac{\sum_{i=0}^{n} y_i \cdot \exp\left(-\frac{1}{2}\left(\frac{x-\frac{i}{n}}{h}\right)^2\right)}{\sum_{i=0}^{n} \exp\left(-\frac{1}{2}\left(\frac{x-\frac{i}{n}}{h}\right)^2\right)} \ .$$

Now the given data are extended from $[0,1]$ to $\mathbb{R}$, by first reflecting the data $y_{-i} = y(-x_i) = y(x_i) = y_i$ so as to obtain data in $[-1,1]$ with $y(-1) = y(1)$, and then by repeating these values periodically. An example is given in Figure 2.1.

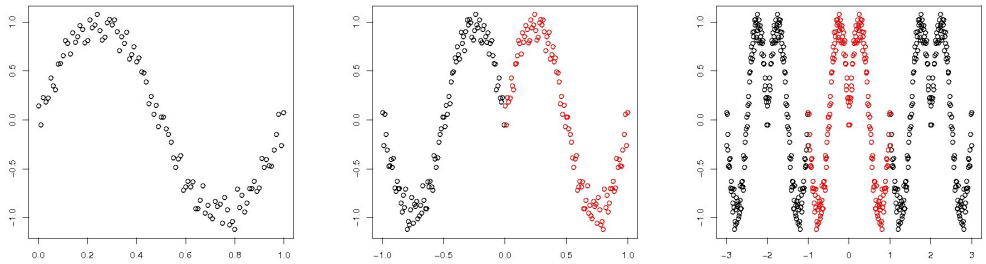


Figure 2.1: Extension of the Sine dataset

The estimator $f_{n,h}^{\infty}$ on $\mathbb{R}$ is defined as

$$f_{n,h}^{\infty}(x) = \frac{\sum_{i=-\infty}^{\infty} y_i \cdot \exp\left(-\frac{1}{2}\left(\frac{x-\frac{i}{n}}{h}\right)^2\right)}{\sum_{i=-\infty}^{\infty} \exp\left(-\frac{1}{2}\left(\frac{x-\frac{i}{n}}{h}\right)^2\right)}.$$

The sums in the numerator and denominator can be multiplied by $\frac{1}{n}$ in order to obtain Riemann sums, and taking the limit as $n \to \infty$, one gets

$$\frac{1}{n}\sum_{i=-\infty}^{\infty} y_i \cdot \exp\left(-\frac{1}{2}\left(\frac{x-\frac{i}{n}}{h}\right)^2\right) \xrightarrow{n\to\infty} \int_{-\infty}^{\infty} y(s) \cdot \exp\left(-\frac{1}{2}\left(\frac{x-s}{h}\right)^2\right) ds \quad \text{and}$$

$$\frac{1}{n}\sum_{i=-\infty}^{\infty} \exp\left(-\frac{1}{2}\left(\frac{x-\frac{i}{n}}{h}\right)^2\right) \xrightarrow{n\to\infty} \int_{-\infty}^{\infty} \exp\left(-\frac{1}{2}\left(\frac{x-s}{h}\right)^2\right) ds,$$

obtaining the smoothing function

$$\hat{f}_h(x) = \frac{\int_{-\infty}^{\infty} y(s) \cdot \exp\left(-\frac{1}{2}\left(\frac{x-s}{h}\right)^2\right) ds}{\int_{-\infty}^{\infty} \exp\left(-\frac{1}{2}\left(\frac{x-s}{h}\right)^2\right) ds}.$$

Substituting $\tau := \frac{h^2}{2}$, the denominator becomes

$$\int_{-\infty}^{\infty} \exp\left(-\frac{1}{2}\left(\frac{x-s}{h}\right)^2\right) ds = \int_{-\infty}^{\infty} \exp\left(-\frac{(x-s)^2}{4\tau}\right) ds = \sqrt{4\pi\tau}$$

and the numerator becomes

$$\int_{-\infty}^{\infty} y(s) \cdot \exp\left(-\frac{1}{2}\left(\frac{x-s}{h}\right)^2\right)\, ds = \int_{-\infty}^{\infty} y(s) \cdot \exp\left(-\frac{(x-s)^2}{4\tau}\right)\, ds,$$

therefore one obtains the integral form of a new estimator $\hat{f}_\tau$ as

$$\hat{f}_\tau(x) = \frac{1}{\sqrt{4\pi\tau}} \int_{-\infty}^{\infty} y(s) \cdot \exp\left(-\frac{(x-s)^2}{4\tau}\right) ds.$$

This function is the convolution of the fundamental solution of the homogeneous heat equation with the data y. Recall that the homogeneous heat equation is given by

$$\frac{\partial}{\partial t} u(x,t) = \frac{\partial^2}{\partial x^2} u(x,t),$$

and its fundamental solution, which is also called the heat kernel, is

$$K(x,t) = \frac{1}{\sqrt{4\pi t}}\, \exp\left(-\frac{|x|^2}{4t}\right).$$

It can be easily checked that $\hat{f}_\tau = u(\cdot, \tau)$ satisfies the homogeneous heat equation with starting values $y(s)$ and stopping time $t = \tau$.

2.2 The Homogeneous Diffusion Process

In the previous section, it has been shown that the kernel estimator can be transformed into a diffusion process, hence data can be smoothed by diffusion. The data smoothing by diffusion can be understood with the following interpretation: The temperature distribution of a metal rod of length one can be taken as the values of the dataset. As the heat equation acts, the temperature diffuses, i.e. the temperature differences decrease. With advancing time the temperature distribution becomes smoother, until eventually the temperature is constant over the whole rod, the constant temperature value being the mean value of the dataset. This phenomenon is illustrated in Figures 2.2 and 2.3 for the Sine dataset.

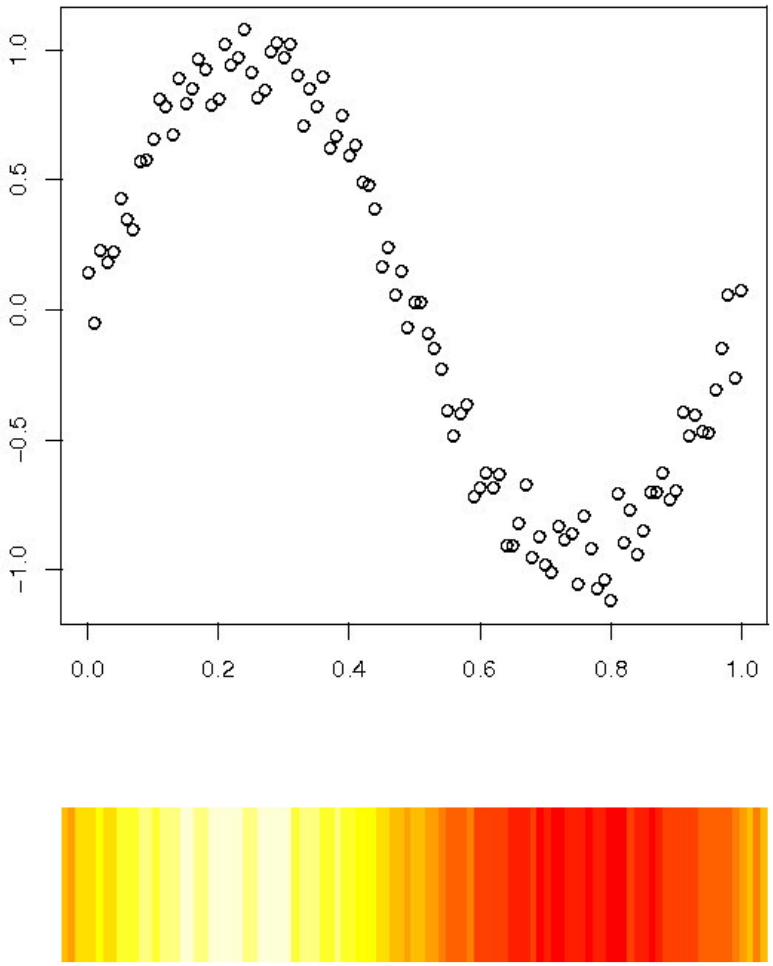


Figure 2.2: Sine dataset and its interpretation as heat distribution

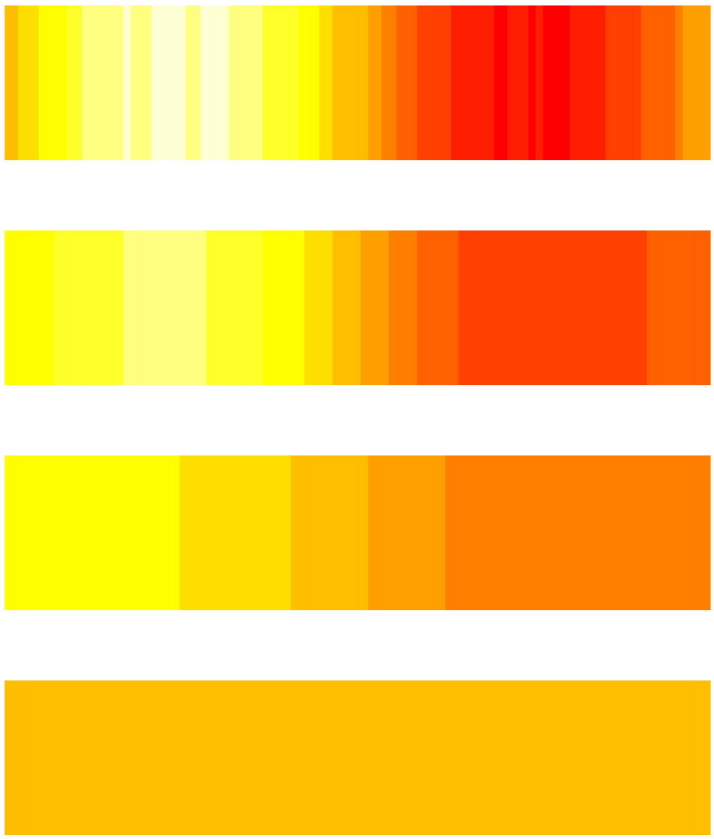


Figure 2.3: Diffusion process stopped at different times $\tau = 0.0001,\ 0.01,\ 0.1,\ 5$ (interpretation as heat distribution)

The advantage of this transformation of the kernel estimator is that the new estimator $\hat{f}_\tau$ is the solution of a partial differential equation, which can be numerically computed much faster than the sums occuring in the kernel estimator. The numerical solution of the diffusion process is presented in Section 2.3.
As the data y on $\mathbb{R}$ are generated by reflection and periodic repetition, a solution of the PDE on $[0,1]$ with homogeneous Neumann boundary conditions, $\frac{\partial u}{\partial x}(0,t) = \frac{\partial u}{\partial x}(1,t) = 0$ for all $t \geq 0$, can be computed instead of the solution on $\mathbb{R}$. The solution on $\mathbb{R}$ is then again obtained by reflection and periodic repetition. In the following, $\hat{f}_\tau|_{[0,1]}$ is the main estimator of interest and for convenience we sometimes abbreviate it as $\hat{f}_\tau$.
The estimator $\hat{f}_\tau$ converges with the optimal rate $n^{-2/5}$. This will be shown in Section 4.1.

2.3 Numerical Solution of the Homogeneous Diffusion Process

Here we consider the homogeneous diffusion process

$$\frac{\partial}{\partial t}u(x,t) = \frac{\partial^2}{\partial x^2}u(x,t)$$

with

- $x \in [0,1],\ t \geq 0$,
- starting values at the design points given by $u(x_i, 0) = y_i$, the remaining starting values $u(x, 0)$ are obtained by interpolation,
- homogeneous Neumann conditions $\frac{\partial u}{\partial x}(0,t) = \frac{\partial u}{\partial x}(1,t) = 0 \ \ \forall t \geq 0$,

and seek for a numerical solution in x_i, $i = 0, \ldots, n$, at the time $t = \tau$. We remark that the x_i need no longer be equidistant points $\frac{i}{n}$, they can be any partition $0 = x_0 < x_1 < \cdots < x_n = 1$ of the interval $[0,1]$. However in order to keep the notation simple, we will only consider equidistant partitions $x_i = \frac{i}{n}$, enabling us to write $\Delta x = \frac{1}{n}$.

First of all the PDE needs to be discretized. As one is interested in the solution in $x_i = i\Delta x$, Δx is used as the location discretization step. Furthermore one is interested in the solution at the stopping time τ, thus if $\Delta t = \tau$ is taken as the time discretization step, only one time step needs to be computed.
In the following, we present one possible discretization and the corresponding scheme for the numerical solution of the heat equation. In order to be flexible in the choice of the smoothing parameter τ, we have used an implicit scheme, ensuring unconditional stability. Note that the explicit scheme is only stable for $\frac{\Delta t}{\Delta x^2} \leq \frac{1}{2}$. In our case Δx is fixed and we wish to use $\Delta t = \tau$. Therefore any constraint on Δt is to be avoided.
For a detailed exposition of the numerical solution of partial differential equations and the stability analysis of the different schemes, we refer to Tveito and Winther [TW98] and Knabner and Angermann [KA00].
The deviations $\frac{\partial}{\partial t}u(x,t)$ and $\frac{\partial^2}{\partial x^2}u(x,t)$ can be discretized using the Taylor expansions of $u(x,t)$ around x and t_0 respectively.

$$u(x, t_0 + \Delta t) = u(x, t_0) + \Delta t \cdot \frac{\partial u}{\partial t}(x, t_0 + \Delta t) + o(\Delta t), \quad \text{hence}$$

$$\frac{\partial u}{\partial t}(x, t_0 + \Delta t) = \frac{u(x, t_0 + \Delta t) - u(x, t_0)}{\Delta t} + o(\Delta t) \approx \frac{u(x, t_0 + \Delta t) - u(x, t_0)}{\Delta t}.$$

For $t_0 = 0$ and $\Delta t = \tau$ the deviation can thus be approximated by the difference quotient

$$\frac{\partial u}{\partial t}(x,\tau) \approx \frac{u(x,\tau) - u(x,0)}{\tau}.$$

For the location discretization both the forward and backward Taylor expansions are used:

$$u(x+\Delta x,\tau) = u(x,\tau) + \Delta x \cdot \frac{\partial u}{\partial x}(x,\tau) + \frac{1}{2}\Delta x^2 \cdot \frac{\partial^2 u}{\partial x^2}(x,\tau) + o(\Delta x^2)$$

$$u(x-\Delta x,\tau) = u(x,\tau) - \Delta x \cdot \frac{\partial u}{\partial x}(x,\tau) + \frac{1}{2}\Delta x^2 \cdot \frac{\partial^2 u}{\partial x^2}(x,\tau) + o(\Delta x^2).$$

Adding them one gets

$$u(x+\Delta x,\tau) + u(x-\Delta x,\tau) = 2u(x,\tau) + \Delta x^2 \cdot \frac{\partial^2 u}{\partial x^2}(x,\tau) + o(\Delta x^2),$$

which yields

$$\frac{\partial^2 u}{\partial x^2}(x,\tau) = \frac{u(x+\Delta x,\tau) - 2u(x,\tau) + u(x-\Delta x,\tau)}{\Delta x^2} + o(\Delta x^2),$$

and the deviation can be approximated by the difference quotient

$$\frac{\partial^2 u}{\partial x^2}(x,\tau) \approx \frac{u(x+\Delta x,\tau) - 2u(x,\tau) + u(x-\Delta x,\tau)}{\Delta x^2}.$$

One obtains the following implicit finite differences scheme for the heat equation:

$$\frac{u(x_i,\tau) - u(x_i,0)}{\tau} \approx \frac{u(x_i-\Delta x,\tau) - 2u(x_i,\tau) + u(x_i+\Delta x,\tau)}{\Delta x^2} \quad \text{or}$$

$$u(x_i,\tau) \approx u(x_i,0) + \frac{\tau}{\Delta x^2} \cdot \left(u(x_{i-1},\tau) - 2u(x_i,\tau) + u(x_{i+1},\tau)\right), \quad i = 1,\dots n-1.$$

Note that for $i = 0$ and $i = n$, it follows from the homogeneous Neumann boundary conditions that $u(x_0 - \Delta x, t) \approx u(x_0 + \Delta x, t)$ and $u(x_n + \Delta x, t) \approx u(x_n - \Delta x, t)$. If the backward Taylor expansion is subtracted from the forward expansion, one gets

$$u(x+\Delta x,\tau) - u(x-\Delta x,\tau) \approx 2\Delta x \frac{\partial u}{\partial x}(x,\tau).$$

The Neumann boundary conditions yield $\frac{\partial u}{\partial x}(x_0,\tau) = \frac{\partial u}{\partial x}(x_n,\tau) = 0$ and thus

$$u(x_0 - \Delta x,\tau) \approx u(x_0 + \Delta x,\tau) = u(x_1,\tau) \quad \text{and}$$

$$u(x_n + \Delta x,\tau) \approx u(x_n - \Delta x,\tau) = u(x_{n-1},\tau).$$

This observation also explains why the reflection of the data corresponds to the homogeneous Neumann boundary conditions.

In order to solve the discrete version of the heat equation one needs to solve the following system of linear equations:

$$\left|\begin{array}{lcl} u(x_0,\tau) & = & u(x_0,0)+\frac{\tau}{\Delta x^2}\cdot(-2u(x_0,\tau)+2u(x_1,\tau)) \\ u(x_1,\tau) & = & u(x_1,0)+\frac{\tau}{\Delta x^2}\cdot(u(x_0,\tau)-2u(x_1,\tau)+u(x_2,\tau)) \\ u(x_2,\tau) & = & u(x_2,0)+\frac{\tau}{\Delta x^2}\cdot(u(x_1,\tau)-2u(x_2,\tau)+u(x_3,\tau)) \\ & \vdots & \\ u(x_{n-1},\tau) & = & u(x_{n-1},0)+\frac{\tau}{\Delta x^2}\cdot(u(x_{n-2},\tau)-2u(x_{n-1},\tau)+u(x_n,\tau)) \\ u(x_n,\tau) & = & u(x_n,0)+\frac{\tau}{\Delta x^2}\cdot(2u(x_{n-1},\tau)-2u(x_n,\tau)) \end{array}\right|$$

i.e. the equation $\mathbf{A}x = \mathbf{b}$ with

$$\mathbf{A} = \begin{pmatrix} 1+2\frac{\tau}{\Delta x^2} & -2\frac{\tau}{\Delta x^2} & 0 & \cdots & \cdots & 0 \\ -\frac{\tau}{\Delta x^2} & 1+2\frac{\tau}{\Delta x^2} & -\frac{\tau}{\Delta x^2} & 0 & & \vdots \\ 0 & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & 0 \\ \vdots & & 0 & -\frac{\tau}{\Delta x^2} & 1+2\frac{\tau}{\Delta x^2} & -\frac{\tau}{\Delta x^2} \\ 0 & \cdots & \cdots & 0 & -2\frac{\tau}{\Delta x^2} & 1+2\frac{\tau}{\Delta x^2} \end{pmatrix},$$

$$\mathbf{b} = \begin{pmatrix} u(x_0,0) \\ u(x_1,0) \\ u(x_2,0) \\ \vdots \\ u(x_n,0) \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \quad \text{and} \quad x = \begin{pmatrix} u(x_0,\tau) \\ u(x_1,\tau) \\ u(x_2,\tau) \\ \vdots \\ u(x_n,\tau) \end{pmatrix} = \begin{pmatrix} \hat{f}_\tau(x_0) \\ \hat{f}_\tau(x_1) \\ \hat{f}_\tau(x_2) \\ \vdots \\ \hat{f}_\tau(x_n) \end{pmatrix}.$$

As $\mathbf{A}$ is a tridiagonal matrix, this equation can be solved exactly by the Gaussian algorithm with $O(n)$ computations. Note that the storage complexity of the algorithm is also of order $O(n)$. See Schwarz [Sch93] for an explanation of the algorithm.

The computing time is independent of the choice of the smoothing parameter τ, whereas the computation of the kernel estimator takes more time with increasing smoothing parameter h. In Table 2.1 the computing time is presented for the kernel estimator with Epanechnikov kernel and the diffusion processs. Different smoothing parameters are used for the Sine dataset of size 10^4. Although the

smoothing parameter is local here, we do not use a fast algorithm for the kernel estimator, but one which allows local bandwidths as well. The diffusion process is computed considerably faster than the kernel estimator. The factor of the speedup depends on the value of the smoothing parameter.

smoothing parameter	kernel estimator	diffusion process
10	13.6 sec.	0.01 sec.
1	13.6 sec.	0.01 sec.
0.1	6.3 sec.	0.01 sec.
0.01	4.8 sec.	0.01 sec.
0.001	4.7 sec.	0.01 sec.
0.0001	4.6 sec.	0.01 sec.

Table 2.1: Computing times of the kernel estimator and the diffusion process for the Sine dataset of size 10^4

2.4 The Inhomogeneous Diffusion Process

Estimators with a single global smoothing parameter are not suitable for datasets with varying smoothness as seen in Section 1.3. The estimator $\hat{f}_\tau$ has such a global smoothing parameter τ. So the next aim is to find a local version of this smoothing parameter. τ has the interpretation as the stopping time of the diffusion process, which cannot be localized. At this point a new parameter a, the so called diffusivity or coefficient of thermal conductivity, can be introduced. The diffusivity controls the velocity of the diffusion.
In what follows the diffusivity a is regarded as a new smoothing parameter and the diffusion process is always stopped at the time $\tau = 1$. The advantage of the diffusivity is that it can be localized. Thus a local smoothing parameter is realized by the local diffusivity $a(x)$. The inhomogeneous diffusion process

$$\frac{\partial}{\partial t}u(x,t) = a(x)\,\frac{\partial^2}{\partial x^2}u(x,t)$$

is considered with

- $x \in [0,1],\ t \geq 0,\ a(x) \geq 0,$
- starting values at the design points given by $u(x_i, 0) = y_i$, the remaining starting values $u(x, 0)$ are obtained by interpolation,
- homogeneous Neumann conditions $\frac{\partial u}{\partial x}(0,t) = \frac{\partial u}{\partial x}(1,t) = 0 \ \ \forall t \geq 0.$

The solution of the inhomogeneous diffusion process can be written in the integral form

$$\hat{f}_{a,\tau}(x) = \frac{1}{\sqrt{4\pi a(x)\tau}} \int_{-\infty}^{\infty} y(s) \cdot \exp\left(-\frac{(x-s)^2}{4a(x)\tau}\right) ds.$$

If it is stopped at $\tau = 1$, one gets the estimator $\hat{f}_a = \hat{f}_{a,\tau=1}$ as

$$\hat{f}_a(x) = \frac{1}{\sqrt{4\pi a(x)}} \int_{-\infty}^{\infty} y(s) \cdot \exp\left(-\frac{(x-s)^2}{4a(x)}\right) ds.$$

For a global diffusivity $a(x) \equiv a$, the value of the diffusivity has the same effect as the stopping time in the homogeneous diffusion process, i.e. $\hat{f}_a = \hat{f}_\tau$ for $a(x) \equiv \tau$. For instance, the diffusivity $a = 2$ doubles the velocity of the diffusion. The solution of the inhomogeneous diffusion process with $a(x) \equiv 2$, stopped at τ, is the same as the solution of the homogeneous diffusion process stopped at 2τ.
Another version of the inhomogeneous diffusion process is given by

$$\frac{\partial}{\partial t} u(x,t) = \frac{\partial}{\partial x}\left(a(x)\frac{\partial}{\partial x} u(x,t)\right),$$

where the diffusivity $a(x)$ appears inside the first derivative $\frac{\partial}{\partial x}$. We denote the solution of this diffusion process stopped at $\tau = 1$ by $\hat{f}_a^{\star}$.
In the above mentioned interpretation regarding the metal rod, the local diffusivity may be thought of representing the composure of the rod of different materials with varying capacity of heat conduction.
One should note here that the new estimator can no longer be interpreted as a kernel estimator. One difference is that one can put $a(x) = 0$. This serves as an isolator and separates the parts at the left and right side of x, and hence there is no smoothing at x. As can be seen from Figure 2.4, the estimators $\hat{f}_a$ and $\hat{f}_a^{\star}$ are capable of reconstructing a sharp edge as the one occuring in the Jump dataset (cf. Appendix A.1.2) at $x = 0.5$. The diffusivity is set equal to 0 at the points of discontinuity and equal to a high value at the remaining places for both versions. Here the version $\hat{f}_a^{\star}$ is to be preferred to $\hat{f}_a$.

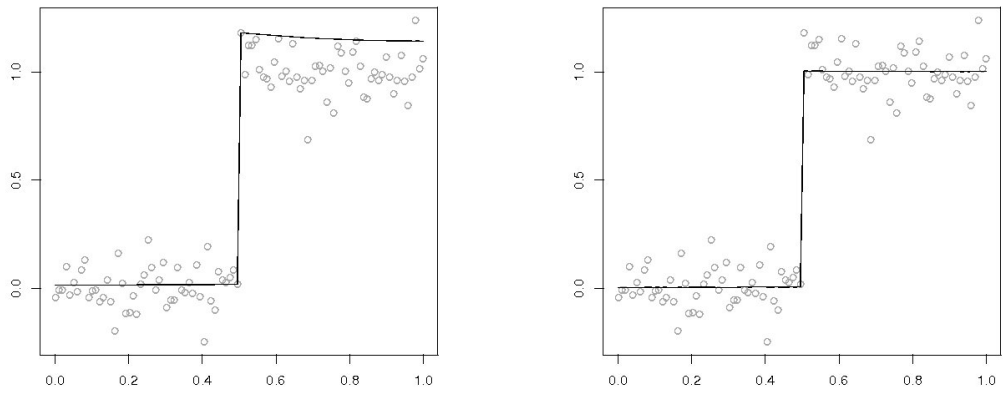


Figure 2.4: Estimators $\hat{f}_a$ (left) and $\hat{f}_a^\star$ (right) for the Jump dataset

Such a sharp edge cannot be achieved by a kernel estimator. A bandwidth $h(x) = 0$ is not possible. Moreover even for a very small value $h(x)$ at the jump, a large bandwidth at the left and the right of the jump result in a smoothing across the jump (cf. Figure 2.5).

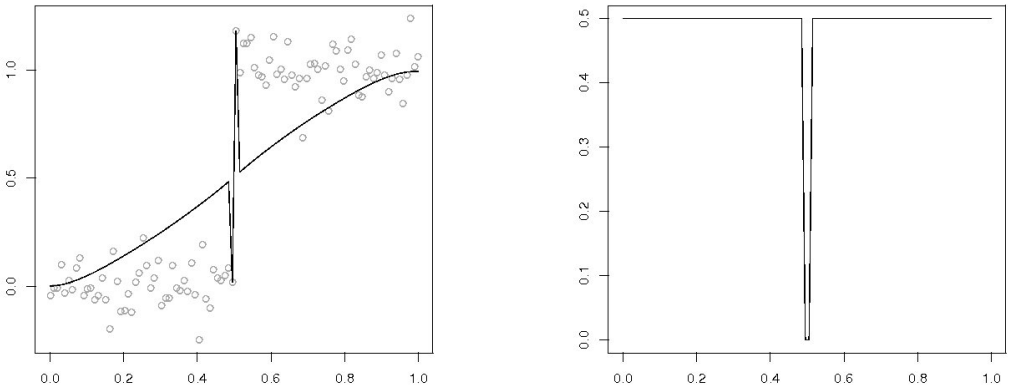


Figure 2.5: Kernel estimator $f_{n,h}$ for the Jump dataset with corresponding bandwidth $h(x)$

If one estimates the two parts of the Jump dataset separately by a kernel estimator with large bandwidth $h \equiv 0.5$, a good approximation can be achieved (cf. Figure 2.6).

This feature is another advantage of the diffusion process over the kernel estimator, in addition to the reduced computing time. It can be utilized if one has information about the discontinuities in the dataset. But in this work we will restrict ourselves to smooth datasets. For datasets containing discontinuities, in particular for datasets which arise from a piecewise constant function, one should choose another smoothing technique, e.g. the total variation minimization, the taut string method or the adaptive weights smoothing, which provide a piecewise constant approximation.

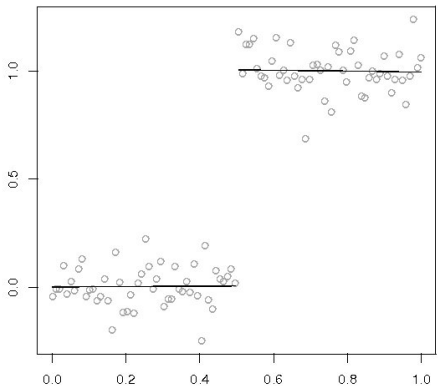


Figure 2.6: Two separate kernel estimators

For smooth datasets, the results for $\hat{f}_a^{\star}$ might be worse than those for $\hat{f}_a$. Figure 2.7 shows both versions with an automatically selected local diffusivity $a(x)$ for the Sine dataset. The selection of the diffusivity is presented in Section 2.6.
As this thesis focuses on smooth datasets, from now on we use $\hat{f}_a$ rather than $\hat{f}_a^{\star}$.

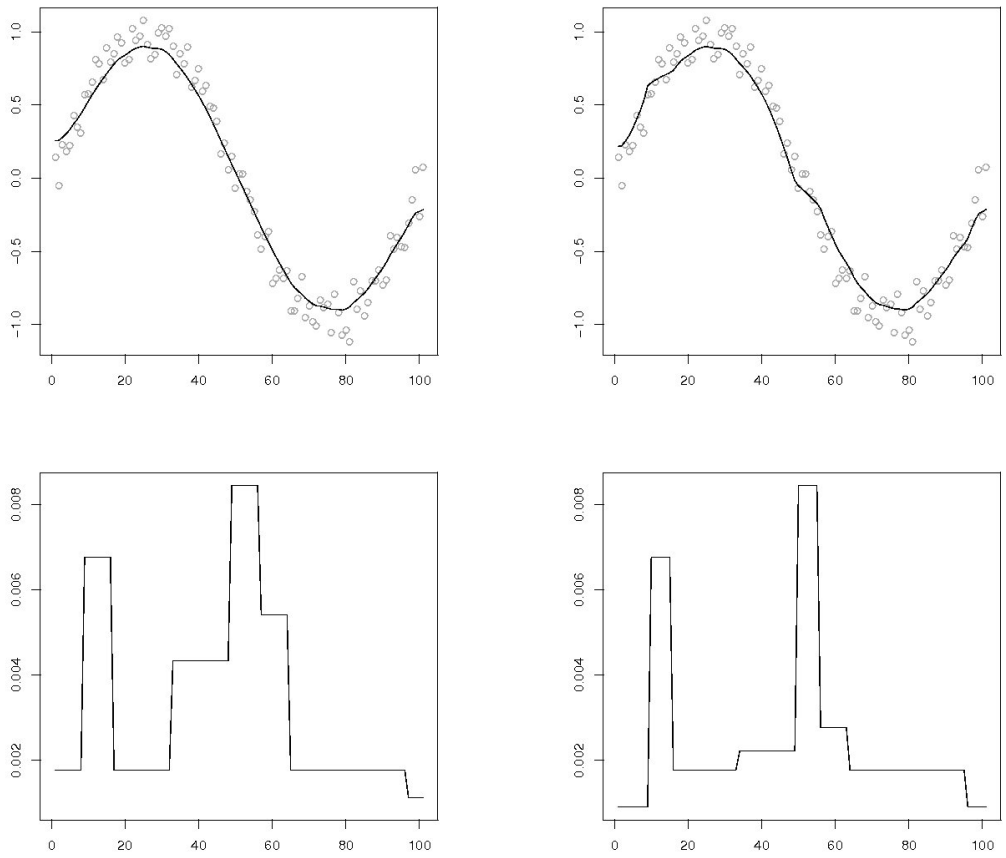


Figure 2.7: Estimators $\hat{f}_a$ (left side) and $\hat{f}_a^{\star}$ (right side) for the Sine dataset Bottom: corresponding selected diffusivities $a(x)$

2.5 Numerical Solution of the Inhomogeneous Diffusion Process

Now we focus on the numerical computation of $\hat{f}_a$. The discrete version of the inhomogeneous diffusion process for $i = 1, \dots n-1$ is

$$\frac{u(x_i,\tau) - u(x_i,0)}{\tau} \approx a(x_i)\left(\frac{u(x_i-\Delta x,\tau) - 2u(x_i,\tau) + u(x_i+\Delta x,\tau)}{\Delta x^2}\right),$$

which yields

$$u(x_i,\tau) \approx u(x_i,0) + \frac{a(x_i)\tau}{\Delta x^2}\cdot\left(u(x_{i-1},\tau) - 2u(x_i,\tau) + u(x_{i+1},\tau)\right).$$

Stopping it at $\tau = 1$, one obtains the following implicit scheme for the estimator $\hat{f}_a$:

$$\hat{f}_a(x_i) = y_i + \frac{a(x_i)}{\Delta x^2} \cdot \left(\hat{f}_a(x_{i-1}) - 2\hat{f}_a(x_i) + \hat{f}_a(x_{i+1}) \right).$$

For $i = 0$ and $i = n$, the homogeneous Neumann boundary conditions are used as in the homogeneous diffusion process. This leads to the tridiagonal linear system $\mathbf{A}x = \mathbf{b}$ with

$$\mathbf{A} = \begin{pmatrix} 1 + \frac{2a(x_0)}{\Delta x^2} & -\frac{2a(x_0)}{\Delta x^2} & 0 & \cdots & \cdots & 0 \\ -\frac{a(x_1)}{\Delta x^2} & 1 + \frac{2a(x_1)}{\Delta x^2} & -\frac{a(x_1)}{\Delta x^2} & 0 & & \vdots \\ 0 & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & 0 \\ \vdots & & 0 & -\frac{a(x_{n-1})}{\Delta x^2} & 1 + \frac{2a(x_{n-1})}{\Delta x^2} & -\frac{a(x_{n-1})}{\Delta x^2} \\ 0 & \cdots & \cdots & 0 & -\frac{2a(x_n)}{\Delta x^2} & 1 + \frac{2a(x_n)}{\Delta x^2} \end{pmatrix},$$

$$\mathbf{b} = \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \quad \text{and } x = \begin{pmatrix} \hat{f}_a(x_0) \\ \hat{f}_a(x_1) \\ \hat{f}_a(x_2) \\ \vdots \\ \hat{f}_a(x_n) \end{pmatrix}.$$

For the version $\hat{f}_a^{\star}$, the right hand side $\frac{\partial}{\partial x}\left(a(x)\,\frac{\partial}{\partial x}u(x,t)\right)$ of the corresponding PDE can be discretized as follows:

$$\frac{\partial}{\partial x}\left(a(x_i)\,\frac{\partial}{\partial x}u(x_i,t)\right) \approx \frac{a(x_{i+\frac{1}{2}})\,\frac{\partial}{\partial x}u(x_{i+\frac{1}{2}},t) - a(x_{i-\frac{1}{2}})\,\frac{\partial}{\partial x}u(x_{i-\frac{1}{2}},t)}{\Delta x}$$

$$\approx \frac{1}{\Delta x}\left(a(x_{i+\frac{1}{2}})\,\frac{u(x_{i+1},t) - u(x_i,t)}{\Delta x} - a(x_{i-\frac{1}{2}})\,\frac{u(x_i,t) - u(x_{i-1},t)}{\Delta x}\right)$$

$$= \frac{\frac{a(x_i)+a(x_{i+1})}{2}\,(u(x_{i+1},t) - u(x_i,t)) - \frac{a(x_{i-1})+a(x_i)}{2}\,(u(x_i,t) - u(x_{i-1},t))}{\Delta x^2}$$

$$= \frac{\frac{a(x_i)+a(x_{i+1})}{2}\,u(x_{i+1},t) - \frac{a(x_{i-1})+2a(x_i)+a(x_{i+1})}{2}\,u(x_i,t) + \frac{a(x_{i-1})+a(x_i)}{2}\,u(x_{i-1},t)}{\Delta x^2}.$$

Writing a_i for $a(x_i)$, one obtains the estimator $\hat{f}_a^{\star}$ by the implicit scheme

$$\hat{f}_a^{\star}(x_i) = y_i + \frac{\frac{a_{i-1}+a_i}{2}\,\hat{f}_a^{\star}(x_{i-1}) - \frac{a_{i-1}+2a_i+a_{i+1}}{2}\,\hat{f}_a^{\star}(x_i) + \frac{a_i+a_{i+1}}{2}\,\hat{f}_a^{\star}(x_{i+1})}{\Delta x^2}$$

for $1 \le i \le n-1$. Using the homogeneous Neumann boundary conditions for $i = 0$ and $i = n$, this leads to the tridiagonal linear system $\mathbf{A}x = \mathbf{b}$ with

$$\mathbf{A} = \begin{pmatrix} 1+\frac{2a_0+2a_1}{2\Delta x^2} & -\frac{2a_0+2a_1}{2\Delta x^2} & 0 & \cdots & \cdots & 0 \\ -\frac{a_0+a_1}{2\Delta x^2} & 1+\frac{a_0+2a_1+a_2}{2\Delta x^2} & -\frac{a_1+a_2}{2\Delta x^2} & 0 & & \vdots \\ 0 & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & 0 \\ \vdots & & 0 & -\frac{a_{n-2}+a_{n-1}}{2\Delta x^2} & 1+\frac{a_{n-2}+2a_{n-1}+a_n}{2\Delta x^2} & -\frac{a_{n-1}+a_n}{\Delta x^2} \\ 0 & \cdots & \cdots & 0 & -\frac{2a_{n-1}+2a_n}{2\Delta x^2} & 1+\frac{2a_{n-1}+2a_n}{2\Delta x^2} \end{pmatrix},$$

$$\mathbf{b} = \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \quad \text{and } x = \begin{pmatrix} \hat{f}_a^{\star}(x_0) \\ \hat{f}_a^{\star}(x_1) \\ \hat{f}_a^{\star}(x_2) \\ \vdots \\ \hat{f}_a^{\star}(x_n) \end{pmatrix}.$$

Note that here the mean values, i.e. $\frac{a_i+a_{i+1}}{2}$, are used instead of a_i (which would be used for the estimator $\hat{f}_a$). This is the reason why $\hat{f}_a^{\star}$ yields better results at discontinuities. It takes into account the neighbouring data points left respectively right from the point of discontinuity, whereas the estimator $\hat{f}_a$ merely keeps the data at the points of discontinuity.

.

2.6 The One-dimensional Multiresolution Criterion

This section is concerned with the selection of the diffusivity $a(x)$ by a statistical analysis of the residuals. One needs a criterion to decide whether the residuals $r_n(x_i)$ of an estimator "look like" Gaussian white noise. For this purpose Davies and Kovac [DK01] have developed the so called multiresolution criterion, which is based on Lévy′s modulus of continuity of the Brownian motion. Here we will

explain the multiresolution criterion and its use in practice. Its theoretical background is given in Chapter 5.
As explained in Section 5.3, the embedding 5.11 together with Theorem 5.3 yields

$$\max_{I\in\mathcal{I}} \frac{\left|\sum_{i\in I} Y_i\right|}{\sqrt{|I|}} \leq \sigma\sqrt{\delta \log N}$$

for an i.i.d. sequence Y_i, $i = 0, \ldots, n$ with $Y_i \sim \mathcal{N}(0, \sigma^2)$ and $N = n + 1$, where $\mathcal{I}$ consists of all the subintervals of $[0, n]$.
This inequality can be used to check whether all the residuals $r_n(x_i) = y_i - \hat{f}_n(x_i)$ of an estimator $\hat{f}_n$ are i.i.d. $\mathcal{N}(0, \sigma^2)$. For this purpose the so called multiresolution coefficients

$$w_I = \frac{1}{\sqrt{|I|}} \sum_{i\in I} r_n(x_i)$$

are considered, and the condition

$$|w_I| \leq \sigma\sqrt{\delta \log N}$$

is checked for all subintervals $I \subseteq [x_0, x_n] = [0, 1]$.
If the residuals fulfill this criterion, they are accepted as Gaussian white noise. Otherwise the residuals contain some further information of the signal and are not accepted as noise, in other words the corresponding $\hat{f}_n$ is not accepted as a good estimator for the data y_i, $i = 0, \ldots, n$. One can even determine the region, where $\hat{f}_n$ is not accepted, namely one can detect the intervals I with $|w_I| > \sigma\sqrt{\delta \log N}$. The constant $\delta > 2$ can be determined by simulations.
The different resolutions or scales in this analysis of the residuals correspond to the size of the intervals I, where naturally the finest scale corresponds to $I = \{x_i\}$ and the coarsest scale corresponds to $I = [0, 1]$.
We refer to Davies et al. [DKM08], where the concept of the multiresolution criterion is formulated in terms of a confidence region.

In practice one does not know the standard deviation σ of the noise and has to estimate it. In what follows the robust estimate σ_n, given by

$$\sigma_n = \frac{1.418}{\sqrt{2}} \text{med}\left(|y_1 - y_0|, \ldots, |y_n - y_{n-1}|\right) \approx \text{med}\left(|y_1 - y_0|, \ldots, |y_n - y_{n-1}|\right),$$

is used.
For a smooth function f, the random variables Z_i with $Z_i \sim \mathcal{N}(0, 1)$ i.i.d. and the data $y_i = f(x_i) + \sigma Z_i$, the differences $y_i - y_{i-1} = f(x_i) - f(x_{i-1}) + \sigma Z_i - \sigma Z_{i-1}$ are close to $\sigma Z_i - \sigma Z_{i-1}$. So one approximately has

$$y_i - y_{i-1} \sim \mathcal{N}(0, 2\sigma^2).$$

The rounded value for the median of $|X|$ with $X \sim \mathcal{N}(0,1)$ is $\frac{1}{1.481}$, hence

$$\text{med } (|y_i - y_{i-1}|) \approx \frac{\sqrt{2}\sigma}{1.481}, \quad i = 1, \ldots, n, \quad \text{which gives}$$

$$\text{med } (|y_1 - y_0|, \ldots, |y_n - y_{n-1}|) \approx \frac{\sqrt{2}\sigma}{1.481}.$$

Therefore

$$\sigma_n = \frac{1.418}{\sqrt{2}} \text{med } (|y_1 - y_0|, \ldots, |y_n - y_{n-1}|)$$

is a good estimate for the standard deviation σ of the noise. Its robustness is inherited from the robustness of the median.

Our aim is to find the largest possible diffusivity $a(x)$, which yields the smoothest estimator $\hat{f}_a$ in the set

$$\mathcal{F} = \left\{ \hat{f}_a \mid \ a(x) \geq 0 : \left| \frac{1}{\sqrt{|I|}} \sum_{i \in I} r_n(x_i) \right| \leq \sigma_n \sqrt{\delta \log N} \quad \forall\, I \in \mathcal{I} \right\},$$

i.e. of all estimators $\hat{f}_a$ satisfying the multiresolution condition for Gaussian white noise.

This is approximately achieved by the following iterative procedure to select the smoothing parameter $a(x)$ of the estimator $\hat{f}_a$:

1. start with a large constant value $a(x_i) = \alpha$ for $i = 0, \ldots, n$,
2. compute $\hat{f}_a$ and the residuals $r_n(x_i)$,
3. check the multiresolution constraint $|w_I| \leq \sigma_n \sqrt{\delta \log N}$ on each interval $I \in \mathcal{I}$,
4. reduce the diffusivity $a(x_i)$ by a factor $\lambda < 1$ at all $x_i \in I$ if the multiresolution constraint is violated on I,
5. repeat steps 2-4 until the multiresolution condition is satisfied on each interval $I \in \mathcal{I}$.

Eventually one obtains approximately the smoothest estimator $\hat{f}_a$ in the set $\mathcal{F}$.

Figure 2.8 shows the estimator $\hat{f}_a$ and its smoothing parameter $a(x)$ for the X-ray Diffractogram dataset. Its computation takes less than half a second for 70 iteration steps (compare with 117 seconds for 36 iteration steps of the kernel estimator).

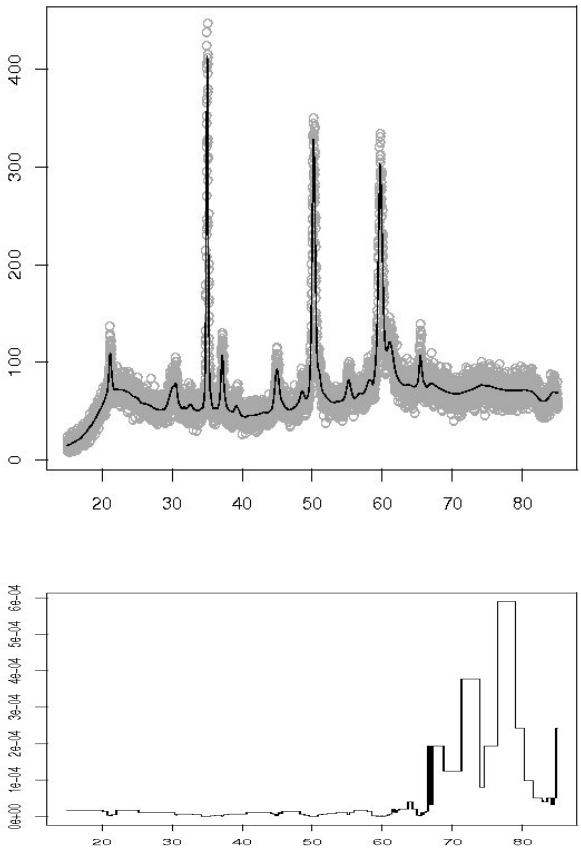


Figure 2.8: Diffusion estimator $\hat{f}_a$ for the X-ray Diffractogram dataset
Bottom: the automatically selected diffusivity $a(x)$

In connection with the iterative selection of the smoothing parameter, we make the following remarks:

- The multiresolution analysis is restricted to the dyadic intervals of the form $[k\,2^j,\ (k+1)\,2^j - 1]$ to speed up the computation. The results are very similar to those obtained when all intervals are considered (cf. Figure 2.9). However, in the case of dyadic intervals one only has to check $O(n)$ intervals instead of $O(n^2)$ intervals.

- We use the default value $\delta = 2.3$ for the multiscale analysis on dyadic intervals.

- As starting value for the diffusivity, the value $\alpha = 25\,\sigma_n/(\max(y_i) - \min(y_i))$ is used.

- The diffusivity is multiplied by $\lambda = 0.8$ in the intervals where the multiresolution constraint is violated.
- Variations of the multiresolution criterion are possible if one uses one of the other inequalities listed in Section 5.3.
- To select the diffusivity a of $\hat{f}_a^\star$ by using the multiresolution condition, we need to decrease not only the values $a(x_i)$ inside the intervals $I = [x_k, x_l]$ where the constraint is violated, but also the neighbouring values $a(x_{k-1})$ and $a(x_{l+1})$. These values have an effect on the values $\hat{f}_a^\star(x_k)$ and $\hat{f}_a^\star(x_l)$, as can be seen from the implicite scheme used for the computation of $\hat{f}_a^\star$.
- The threshold $\sigma\sqrt{\delta \log N}$ with $\delta = 2$ also plays a role in the hard thresholding for wavelets, where $\lambda = \sqrt{2 \log N}$ is given as a universal threshold apart from a minimax threshold, which is tabulated for various sample sizes N. It means that only the wavelet coefficients with an absolute value greater than $\sigma\lambda = \sigma\sqrt{2 \log N}$ are kept.

All the results, that we have shown up to now, have only checked the multiresolution condition on dyadic intervals. See Figure 2.9 for a comparison with the result, where the multiresolution condition is checked on all subintervals. That version takes 47 seconds for 69 iteration steps, and the one where only the dyadic intervals are checked, takes 0.1 second for 67 iteration steps.

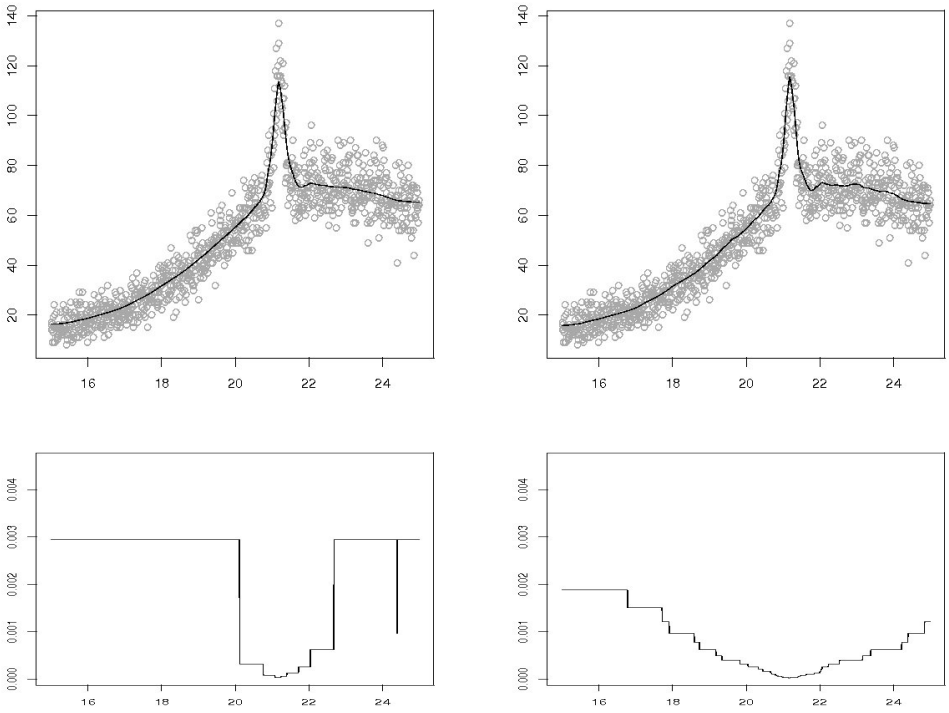


Figure 2.9: Diffusion estimator $\hat{f}_a$ and diffusivity $a(x)$, the multiresolution condition is checked on dyadic intervals (left) and on all intervals (right)

If the multiresolution condition is verified on more intervals, more violations of the constraint may be possible, hence the smoothing parameter is reduced at more places. That is the reason why the version, where all intervals are checked, becomes less smooth. On the other hand, if the condition is checked on a larger set of intervals $\mathcal{I}$, the supremum $\sup_{I \in \mathcal{I}} |\frac{1}{\sqrt{|I|}} r_n(x_i)|$ gets larger, necessitating the use of a larger δ.
In Figure 2.10 the result for $\delta = 5.5$ is shown. This is the value for δ which yields at the same time a smooth result and a reasonable approximation of the peak. Please note that the height of the peak is lower than in the version, where only dyadic intervals are checked. This results from the larger value for δ. The version, which checks only the dyadic intervals, produces smoother results and at the same time is able to approximate small features. Therefore it is preferred, not only for fast computation.

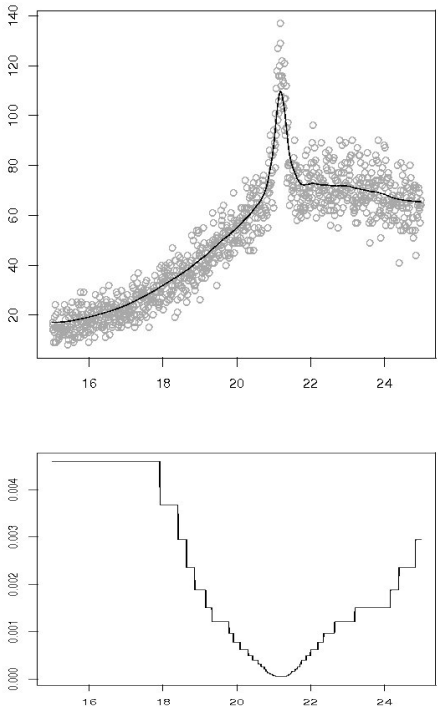


Figure 2.10: Diffusion estimator $\hat{f}_a$ and diffusivity $a(x)$, the multiresolution condition with $\delta = 5.5$ is checked on all intervals

2.7 Numerical Results

The performance of the estimator $\hat{f}_a$, apart from the X-ray Diffractogram dataset, is displayed for the four test signals, that Donoho et al. [DJKP95] consider for their wavelet shrinkage. They are often used to test the performance of an estimator since they contain variant difficulties. In Figures 2.11, 2.12, 2.13 and 2.14 the estimator $\hat{f}_a$ and the selected diffusivity $a(x)$ are plotted for the Bumps dataset, the Heavisine dataset, the Doppler dataset and the Blocks dataset. Here the root signal-to-noise ratio is 5. (cf. Appendix A.1.4.)
The results for datasets like Bumps, Heavisine and Doppler, which are rather smooth, are satisfying, even when discontinuities occur (like for the Heavisine dataset). For the Blocks dataset one could do much better, if the positions of the discontinuities were identified.
In Table 2.2, the median of the mean squared error MSE

$$\frac{1}{n+1}\sum_{i=0}^{n}(f(x_i) - \hat{f}(x_i))^2$$

from 1000 simulations is recorded for different estimators. The lowest values are written in bold face.
The inhomogeneous diffusion estimator $\hat{f}_a$ is computed by the function `diffest1d` (cf. Appendix B.1), the kernel estimator with local plug-in bandwidth is computed by the function `lokerns` (`R`–package `lokern` [HM03]), the taut string is computed by the function `pmreg` (`R`-package `ftnonpar` [DK05a]) and the hard wavelet thresholding is computed by the functions `wd`, `threshold` and `wr` (`R`–package `wavethresh` [NKM06], [Nas93] and [NS94]). We use two different wavelet families: the Daubechies wavelets with least asymmetry, which yield rather smooth solutions, and the Haar wavelets, which perform very well with discontinuous signals. For the wavelet thresholding, the MAD is used to estimate the standard deviation σ of the noise.

	Bumps	Heavisine	Doppler	Blocks
Diffusion estimator $\hat{f}_a$	0.0093	0.0050	0.0143	0.0138
Kernel estimator	0.0112	**0.0020**	0.0083	0.0139
Taut string	**0.0060**	0.0026	0.0081	**0.0013**
Daubechies wavelets	0.0139	0.0030	**0.0037**	0.0177
Haar wavelets	0.0232	0.0078	0.0236	0.0026

Table 2.2: MSE for test signals and different estimators

The kernel estimator with plug-in bandwidth gives good results in terms of the MSE, but we have already discussed in Section 1.2 the problems involved. The MSE is just one value to assess the performance of an estimator, but if we additionally seek for a smooth estimator, the performance can be judged better if one looks at the result itself. Below, in Figures 2.15, 2.16, 2.17 and 2.18 the test signals and the different estimators are plotted.
The Blocks test signal is piecewise constant and contains many discontinuities. Therefore, as expected, the taut string reconstructs the signal in the best way. This can be seen not only in the very low mean value of the MSE, but also in the plot below.
The performance of wavelet thresholding depends on the smoothness of the data and the chosen wavelet family. It produces more artefacts than the other methods.

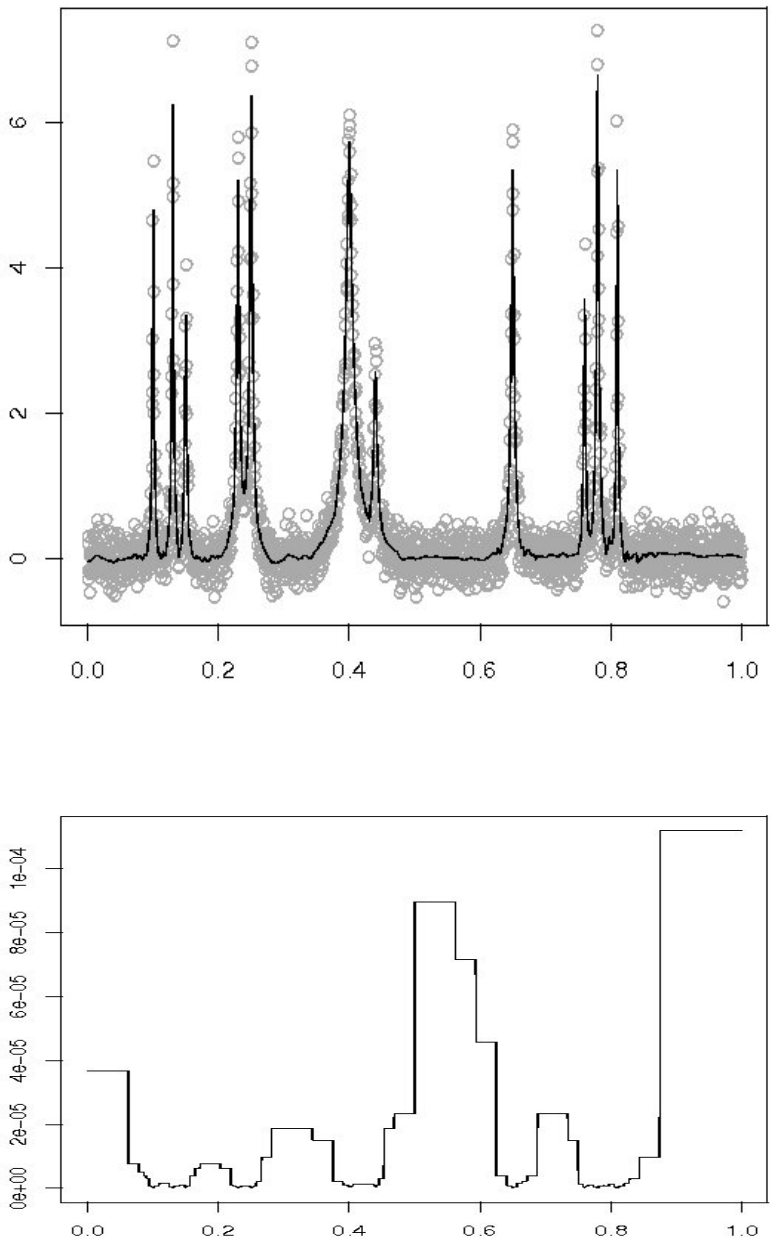


Figure 2.11: $\hat{f}_a$ and the selected diffusivity $a(x)$ for the Bumps dataset

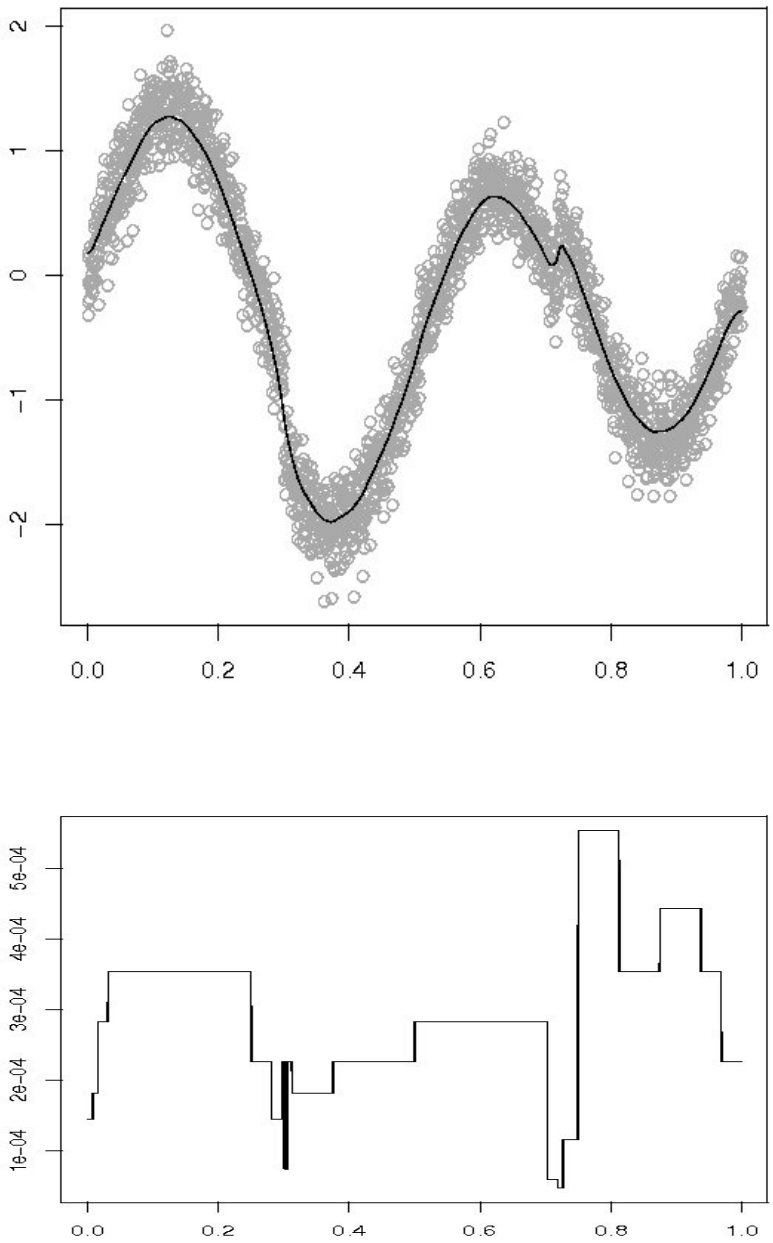


Figure 2.12: $\hat{f}_a$ and the selected diffusivity $a(x)$ for the Heavisine dataset

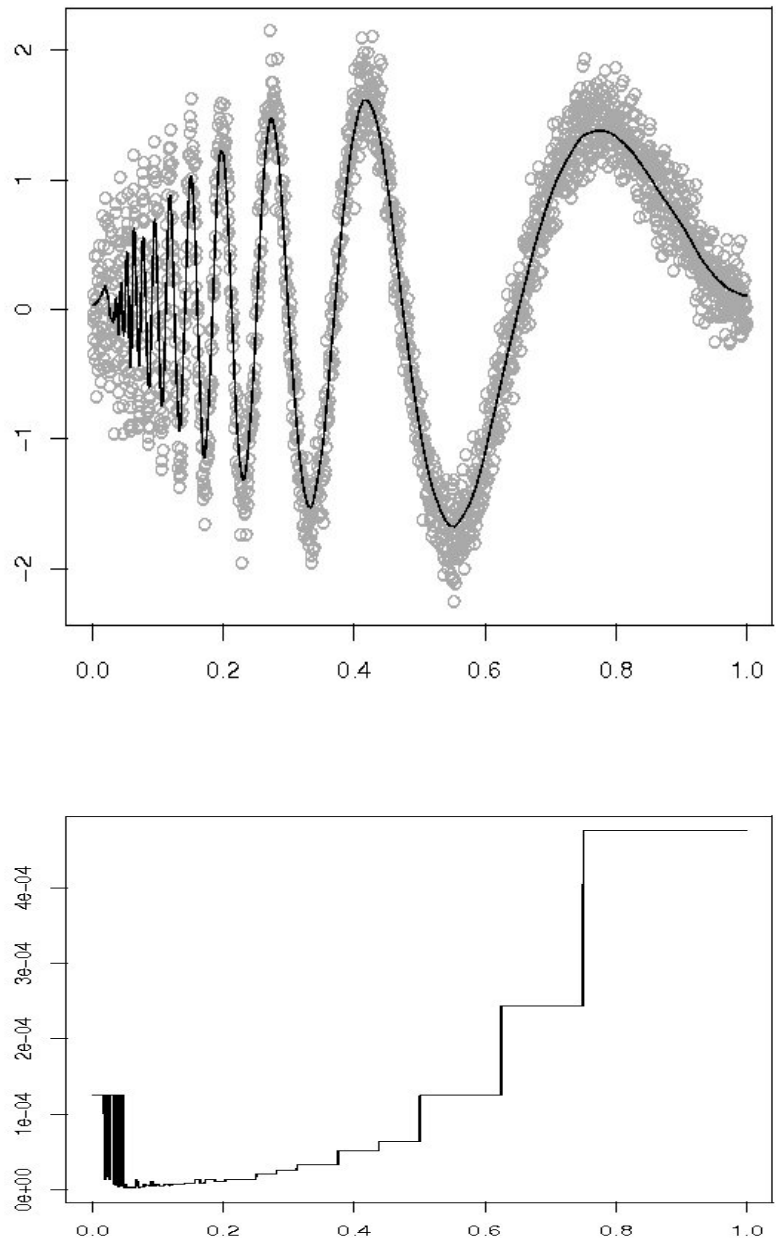


Figure 2.13: $\hat{f}_a$ and the selected diffusivity $a(x)$ for the Doppler dataset

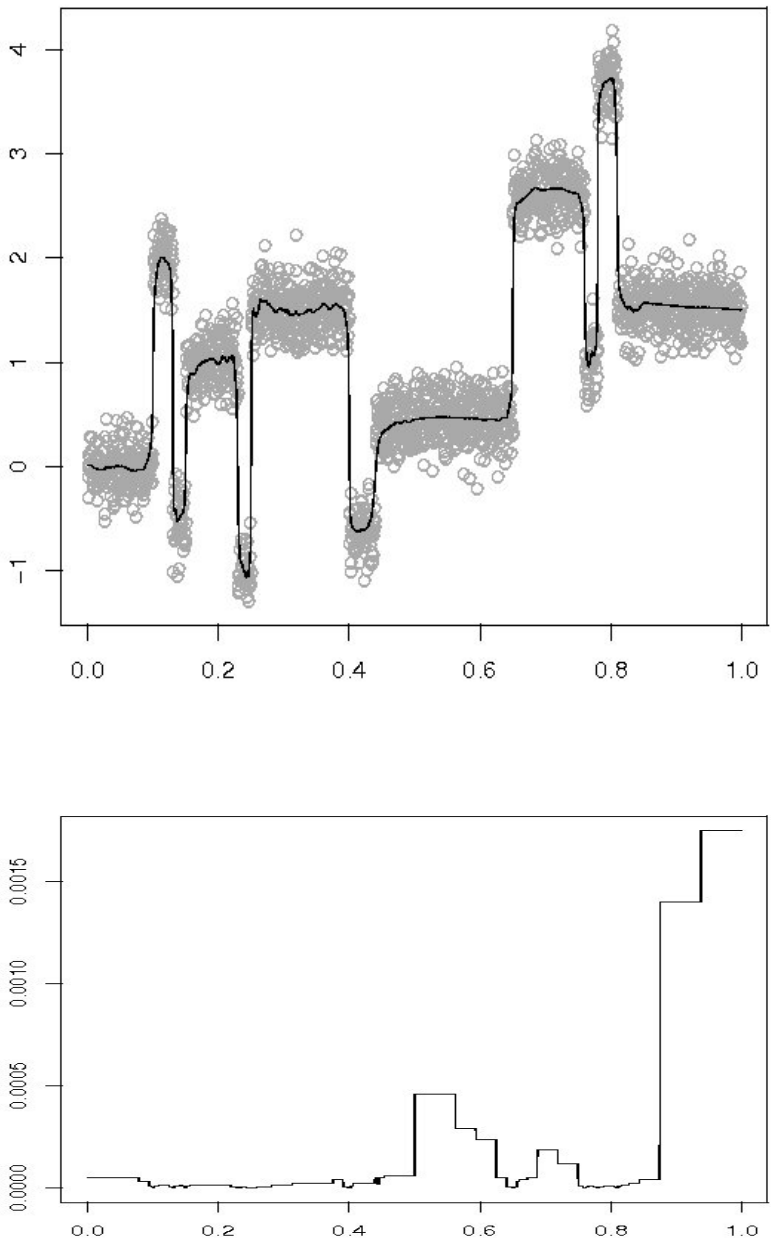
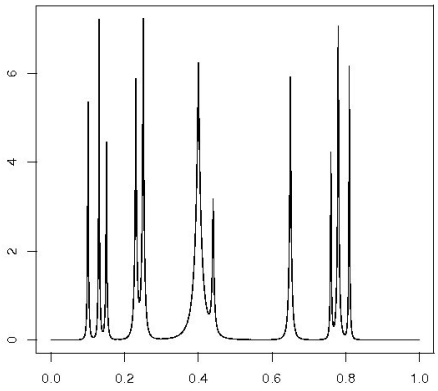
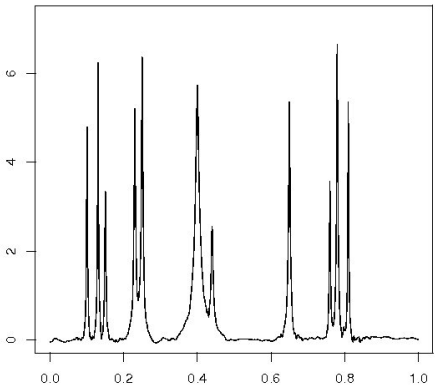


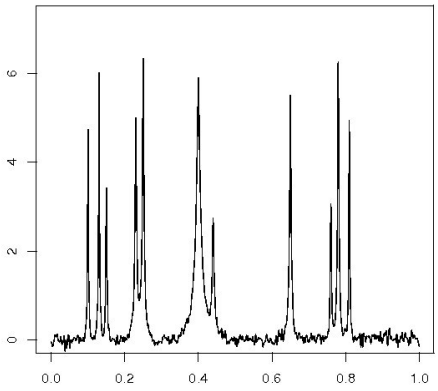
Figure 2.14: $\hat{f}_a$ and the selected diffusivity $a(x)$ for the Blocks dataset



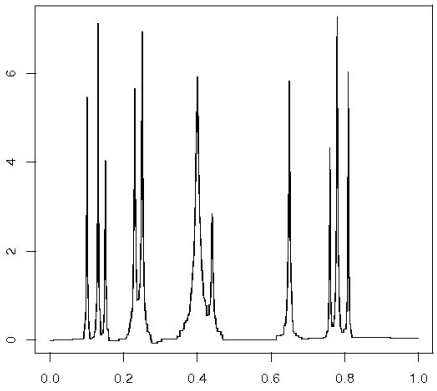
Bumps test signal



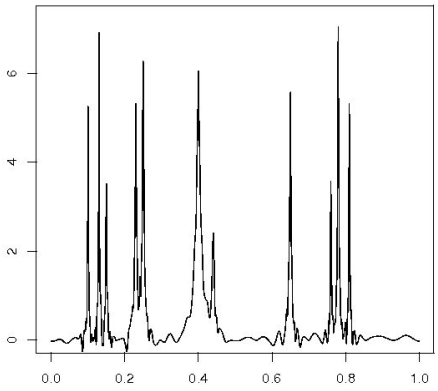
Estimator $\hat{f}_a$



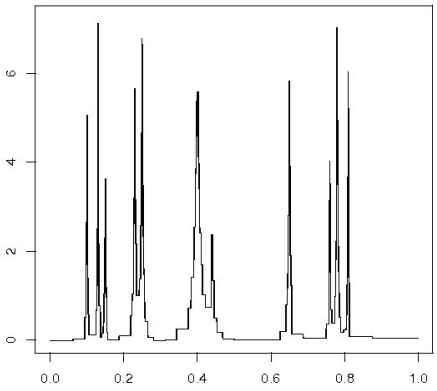
Kernel estimator



Taut string

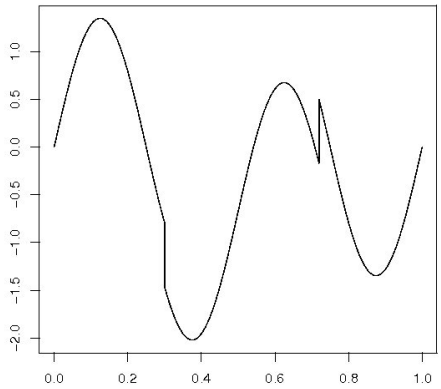


Daubechies wavelets

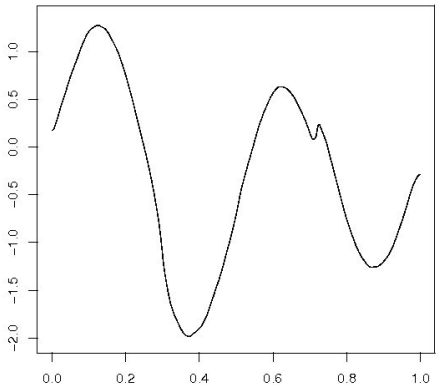


Haar wavelets

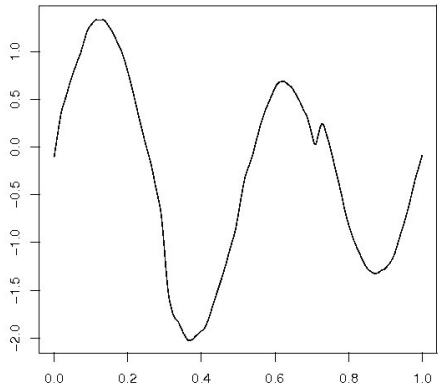
Figure 2.15: Bumps test signal and different estimators



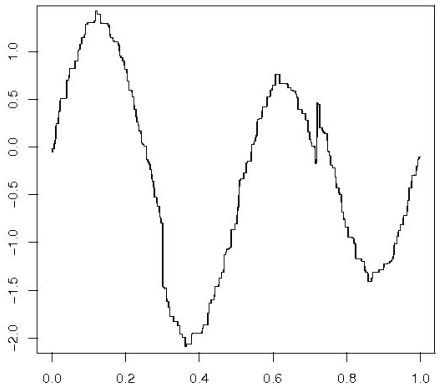
Heavisine test signal



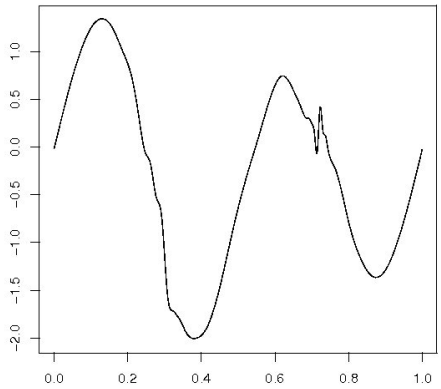
Estimator $\hat{f}_a$



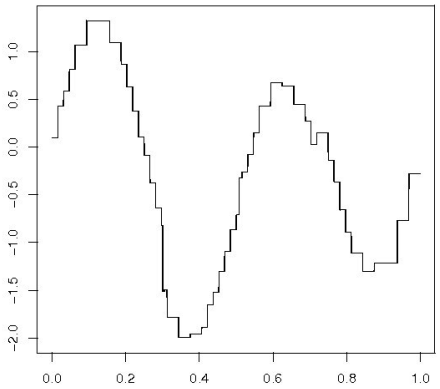
Kernel estimator



Taut string

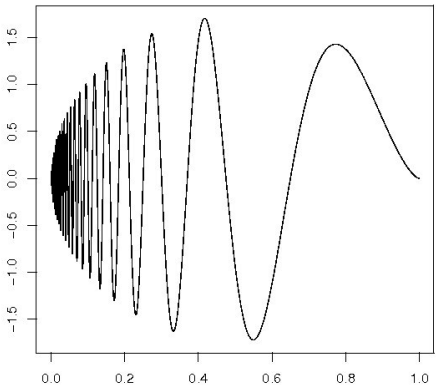


Daubechies wavelets

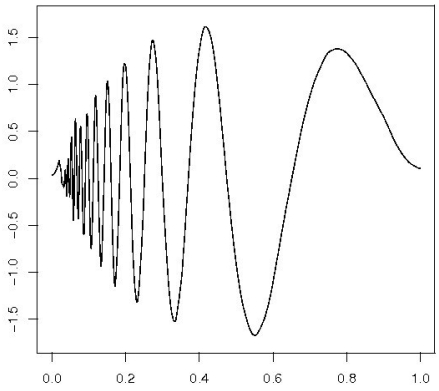


Haar wavelets

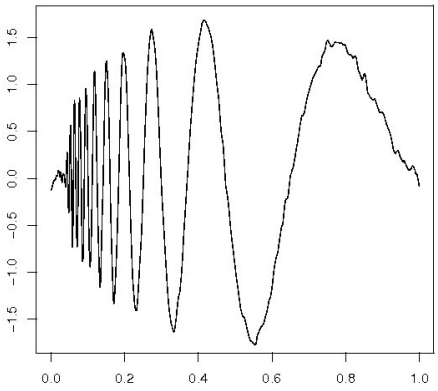
Figure 2.16: Heavisine test signal and different estimators



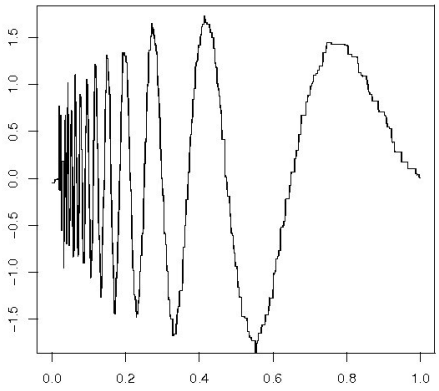
Doppler test signal



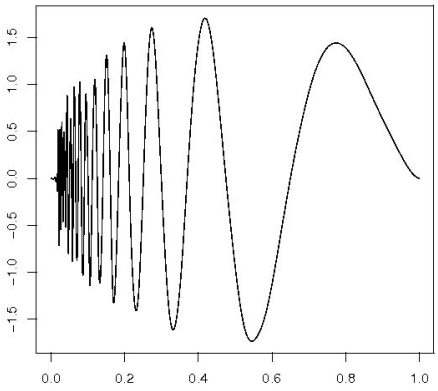
Estimator $\hat{f}_a$



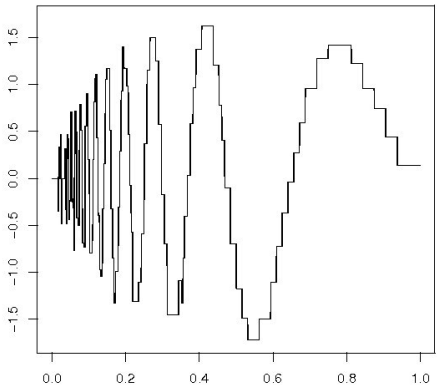
Kernel estimator



Taut string

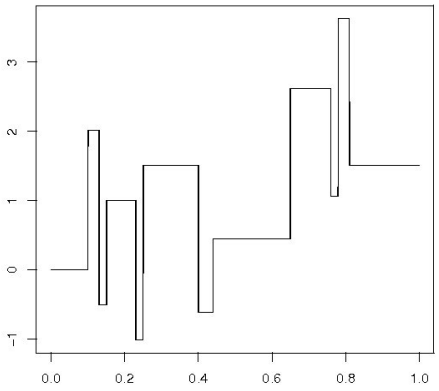


Daubechies wavelets

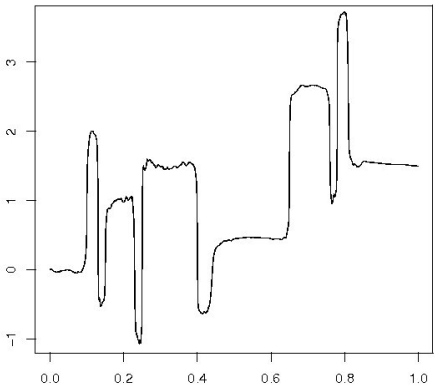


Haar wavelets

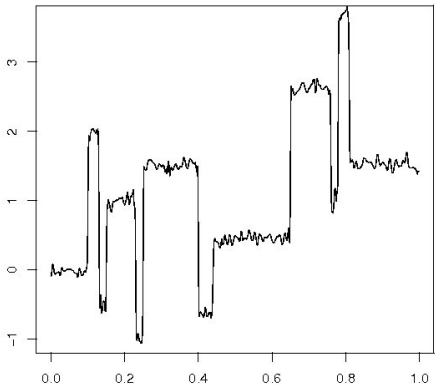
Figure 2.17: Doppler test signal and different estimators



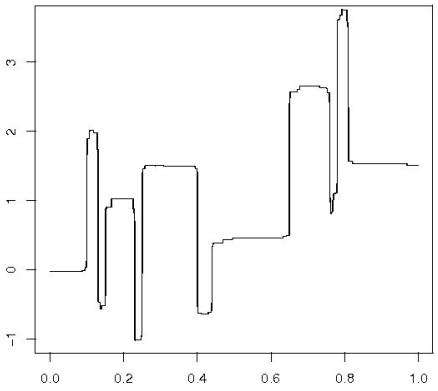
Blocks test signal



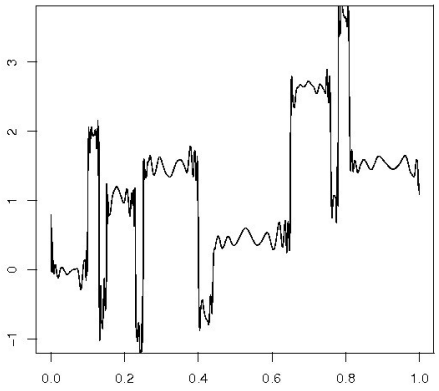
Estimator $\hat{f}_a$



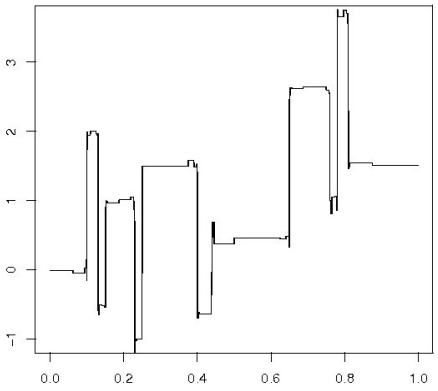
Kernel estimator



Taut string



Daubechies wavelets



Haar wavelets

Figure 2.18: Blocks test signal and different estimators

Chapter 3

The Two-dimensional Diffusion Estimator

In this chapter the method developed in Chapter 2 is applied to images, i.e. to two-dimensional datasets.

3.1 The Model

The two-dimensional nonparametric regression is concerned with datasets of the form

$$(\mathbf{x}_{ij}, y_{ij}),\ i = 0, \dots, m,\ j = 0, \dots, n \quad \text{with } \mathbf{x}_{ij} \in [0,1] \times [0,1] \text{ and } y_{ij} \in \mathbb{R}.$$

For an image with $N = (m+1) \times (n+1)$ equidistant pixels, the points $\mathbf{x}_{ij}$ can also be interpreted as a pair of two natural numbers $\mathbf{x}_{ij} = (m_i, n_j)$ with $m_i = 0, \dots, m$ and $n_i = 0, \dots, n$, so that one has $\mathbf{x}_{ij} \in [0, m] \times [0, n]$. We will always use this interpretation.
The model for the two-dimensional nonparametric regression is as follows:

$$Y_{ij} = f(\mathbf{x}_{ij}) + \sigma\, Z_{ij}, \quad Z_{ij} \text{ i.i.d. } \mathcal{N}(0,1),$$

where σ is the standard deviation of the noise.
We wish to find a representation $\hat{f}(\mathbf{x}_{ij})$ for the given data y_{ij}, hence we look for a decomposition

$$y_{ij} = \hat{f}(\mathbf{x}_{ij}) + r(\mathbf{x}_{ij}),$$

where $\hat{f}$ is an estimator for the data, and the residuals $r(\mathbf{x}_{ij})$ represent the noise.

As an example, we consider the Mexican Hat dataset (cf. Appendix A.2.1).

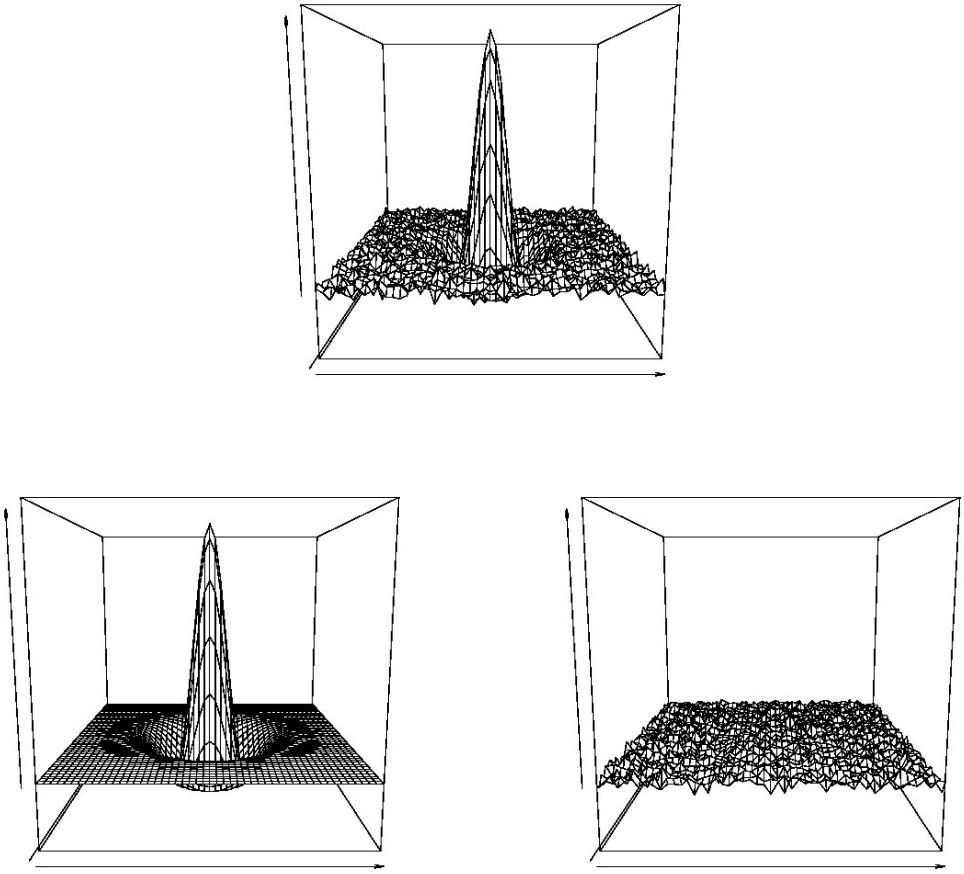


Figure 3.1: Mexican Hat dataset and perfect decomposition into $f(\mathbf{x}_{ij})$ and noise σZ_{ij}

3.2 The Two-dimensional Diffusion Process

Now the diffusion process is used for an approximation of the data y_{ij}. As in one dimension, we consider the homogeneous diffusion process stopped at the time $t = \tau$, resulting in the estimator $\hat{f}_\tau$ with the stopping time τ of the process as a global smoothing parameter, and the inhomogeneous diffusion process stopped at the time $\tau = 1$ resulting in the estimator $\hat{f}_a$ with the diffusivity $a(\mathbf{x})$ as a local smoothing parameter.

3.2.1 The Homogeneous Diffusion Process

The two-dimensional homogeneous diffusion process is given by

$$\frac{\partial}{\partial t}u(\mathbf{x},t) = \Delta u(\mathbf{x},t).$$

As an estimator $\hat{f}_\tau$ for the data, one takes the solution of the homogeneous diffusion process with homogeneous Neumann boundary conditions stopped at the time τ, i.e. $\hat{f}_\tau(\mathbf{x}) = u(\mathbf{x},\tau)$ with

- $\frac{\partial}{\partial t}u(\mathbf{x},t) = \Delta u(\mathbf{x},t)$ for $\mathbf{x} \in \Omega = [0,m] \times [0,n],\ t \geq 0$,
- starting values at the design points given by $u(\mathbf{x}_{ij},0) = y_{ij}$, the remaining starting values $u(\mathbf{x},0)$ are obtained by interpolation,
- Neumann conditions $\nabla u(\mathbf{x},t) = 0$ for $\mathbf{x} \in \partial\,\Omega$.

The starting values of the diffusion process are given by the data.
As in dimension one, the global smoothing parameter of the estimator $\hat{f}_\tau$ is the stopping time τ.

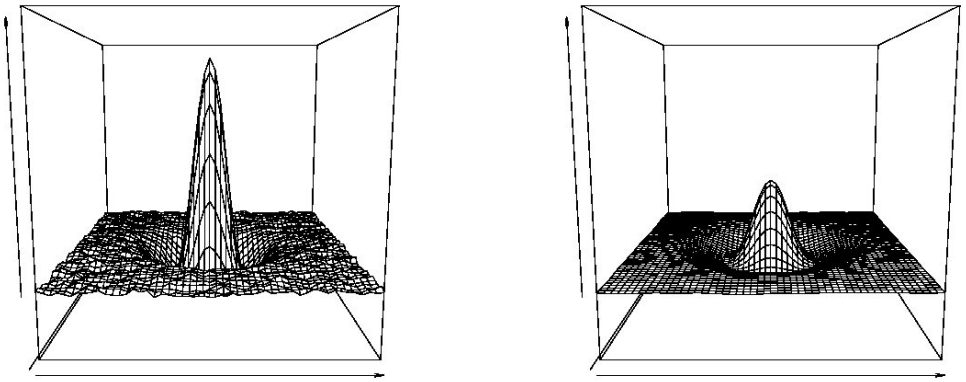


Figure 3.2: Estimator $\hat{f}_\tau$ for the Mexican Hat dataset with different stopping times $\tau = 0.5$ (left) and $\tau = 10$ (right)

Figure 3.2 shows the estimator $\hat{f}_\tau$ for the Mexican Hat dataset. In this example, the problem caused by using a single, global smoothing parameter becomes obvious: We can either get a good approximation of the peak (as on the left) or a smooth approximation (as on the right). This problem can be overcome by a local smoothing parameter as explained in Section 2.4 for the one-dimensional setting.

3.2.2 The Inhomogeneous Diffusion Process

A local smoothing parameter can be achieved by introducing the diffusivity $a(\mathbf{x})$. The two-dimensional inhomogeneous diffusion process is given by

$$\frac{\partial}{\partial t}u(\mathbf{x},t) = a(\mathbf{x})\ \Delta u(\mathbf{x},t), \quad \mathbf{x} \in \Omega = [0,m] \times [0,n],\ t \geq 0,\ a(\mathbf{x}) \geq 0.$$

Hence the estimator $\hat{f}_a$ with the local smoothing parameter $a(\mathbf{x})$ can be defined as the solution $\hat{f}_a(\mathbf{x}) = u(\mathbf{x},1)$ of the inhomogeneous diffusion process stopped at $\tau = 1$ with

- diffusivity $a(\mathbf{x}) \geq 0$,
- starting values at the design points given by $u(\mathbf{x}_{ij},0) = y_{ij}$, the remaining starting values $u(\mathbf{x},0)$ are obtained by interpolation,
- homogeneous Neumann conditions $\nabla u(\mathbf{x},t) = 0$ for $\mathbf{x} \in \partial\Omega$.

3.2.3 Numerical Solution of the Two-dimensional Diffusion Process

In the following we present the numerical solution of the two-dimensional inhomogeneous diffusion process, which is needed to compute the estimator $\hat{f}_a$. For a constant diffusivity $a(\mathbf{x}) \equiv \tau$, this becomes the estimator $\hat{f}_\tau$.
For the numerical solution of a partial differential equation, the PDE is first discretized in order to obtain a system of linear equations. In the next step, the system of equations has to be solved.
There are mainly two methods to discretize partial differential equations: the finite differences method and the finite elements method. In image analysis, our data are given on a very regular grid: for each pixel, we have a data point. We are interested in the solution at each pixel. Thus, the pixel grid is a natural grid for the discretization of the PDE, the discretization steps in x- and y-direction are given as $\Delta x = \Delta y = 1$. As the estimator $\hat{f}_a$ is the solution of the inhomogeneous diffusion process stopped at $\tau = 1$, we choose $\Delta t = 1$ as the time discretization step.
Having such a regular grid, we can use the finite differences method, which is much simpler and more comprehensive than the finite elements method.
For $\frac{\partial}{\partial t}u(\mathbf{x},t)$ we use the same discretization as in the one-dimensional case:

$$\frac{\partial u}{\partial t}(\mathbf{x}, t_0 + \Delta t) \approx \frac{u(\mathbf{x}, t_0 + \Delta t) - u(\mathbf{x}, t_0)}{\Delta t} \quad \text{, which gives}$$

$$\frac{\partial u}{\partial t}(\mathbf{x}, 1) \approx u(\mathbf{x}, 1) - u(\mathbf{x}, 0) \quad \text{for } t_0 = 0,\ \Delta t = 1.$$

Now we discretize the Laplace operator Δ:

$$\Delta u(\mathbf{x}_{ij},t) = \frac{\partial^2 u}{\partial x^2}(\mathbf{x}_{ij},t) + \frac{\partial^2 u}{\partial y^2}(\mathbf{x}_{ij},t)$$

$$\approx \frac{u\big(\mathbf{x}_{ij}+(\Delta x,0),t\big) - 2u\big(\mathbf{x}_{ij},t\big) + u\big(\mathbf{x}_{ij}-(\Delta x,0),t\big)}{\Delta x^2}$$

$$+\frac{u\big(\mathbf{x}_{ij}+(0,\Delta y),t\big) - 2u\big(\mathbf{x}_{ij},t\big) + u\big(\mathbf{x}_{ij}-(0,\Delta y),t\big)}{\Delta y^2},$$

and thus for $t=1$, $\Delta x = \Delta y = 1$

$$\Delta u(\mathbf{x}_{ij},1) \approx u(\mathbf{x}_{i+1,j},1) + u(\mathbf{x}_{i-1,j},1) + u(\mathbf{x}_{i,j+1},1) + u(\mathbf{x}_{i,j-1},1) - 4u(\mathbf{x}_{ij},1).$$

Hence the stencil for the Laplace operator Δ is

$$\begin{bmatrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{bmatrix},$$

and the stencil for the operator $a\Delta$, which is used in the inhomogeneous diffusion process, is

$$\begin{bmatrix} & a & \\ a & -4a & a \\ & a & \end{bmatrix}.$$

Therefore one obtains the discrete version of the inhomogeneous diffusion process

$$u(\mathbf{x}_{ij},1) - u(\mathbf{x}_{ij},0)$$

$$= a(\mathbf{x}_{ij}) \cdot \big(u(\mathbf{x}_{i+1,j},1) + u(\mathbf{x}_{i-1,j},1) + u(\mathbf{x}_{i,j+1},1) + u(\mathbf{x}_{i,j-1},1) - 4u(\mathbf{x}_{ij},1)\big),$$

which yields the implicit scheme

$$u(\mathbf{x}_{ij},1)$$

$$= u(\mathbf{x}_{ij},0) + a(\mathbf{x}_{ij}) \cdot \big(u(\mathbf{x}_{i+1,j},1) + u(\mathbf{x}_{i-1,j},1) + u(\mathbf{x}_{i,j+1},1) + u(\mathbf{x}_{i,j-1},1) - 4u(\mathbf{x}_{ij},1)\big).$$

The estimator $\hat{f}_a$ is given by the values $\hat{f}_a(\mathbf{x}_{ij}) = u(\mathbf{x}_{ij},1)$, which is the diffusion process with starting values $u(\mathbf{x}_{ij},0) = y_{ij}$ and homogeneous Neumann boundary conditions stopped at $\tau = 1$. The homogeneous Neumann boundaries $\frac{\partial u}{\partial x}(\mathbf{x}_{0j},t) = 0$, $\frac{\partial u}{\partial x}(\mathbf{x}_{mj},t) = 0$, $\frac{\partial u}{\partial y}(\mathbf{x}_{i0},t) = 0$ and $\frac{\partial u}{\partial y}(\mathbf{x}_{in},t) = 0$ are used to obtain the values $u(\mathbf{x}_{-1,j},1) \approx u(\mathbf{x}_{1,j},1)$, $u(\mathbf{x}_{m+1,j},1) \approx u(\mathbf{x}_{m-1,j},1)$, $u(\mathbf{x}_{i,-1},1) \approx u(\mathbf{x}_{i,1},1)$ and $u(\mathbf{x}_{i,n+1},1) \approx u(\mathbf{x}_{i,n-1},1)$. Hence the estimator $\hat{f}_a$

can be computed using the implicit scheme

$$\hat{f}_a(\mathbf{x}_{ij}) = \frac{y_{ij} + a(\mathbf{x}_{ij}) \cdot \left(\hat{f}_a(\mathbf{x}_{i-1,j}) + \hat{f}_a(\mathbf{x}_{i+1,j}) + \hat{f}_a(\mathbf{x}_{i,j-1}) + \hat{f}_a(\mathbf{x}_{i,j+1})\right)}{1 + 4a(\mathbf{x}_{ij})}.$$

The scheme results in a system of linear equations $Ax = \mathbf{b}$ with

$$x = \begin{pmatrix} \hat{f}_a(\mathbf{x}_{00}) \\ \hat{f}_a(\mathbf{x}_{01}) \\ \vdots \\ \hat{f}_a(\mathbf{x}_{0n}) \\ \hat{f}_a(\mathbf{x}_{10}) \\ \hat{f}_a(\mathbf{x}_{11}) \\ \vdots \\ \hat{f}_a(\mathbf{x}_{1n}) \\ \vdots \\ \vdots \\ \hat{f}_a(\mathbf{x}_{m0}) \\ \hat{f}_a(\mathbf{x}_{m1}) \\ \vdots \\ \hat{f}_a(\mathbf{x}_{mn}) \end{pmatrix} \quad \text{and} \quad \mathbf{b} = \begin{pmatrix} y_{00} \\ y_{01} \\ \vdots \\ y_{0n} \\ y_{10} \\ y_{11} \\ \vdots \\ y_{1n} \\ \vdots \\ \vdots \\ y_{m0} \\ y_{m1} \\ \vdots \\ y_{mn} \end{pmatrix},$$

i.e. the lth components of x and $\mathbf{b}$ are the values of the estimator $\hat{f}_a$ and the data y, respectively, at the points $\mathbf{x}_{ij}$, where $(n+1) \cdot i + j = l$.
The diagonal elements of the matrix A are given as $A_{ll} = 1 + 4a(\mathbf{x}_{ij})$. Note that the entries A_{lk} are equal to zero except for the ones specified below:

- for the inner points $\mathbf{x}_{ij}$, i.e. for $1 \leq i \leq m-1$ and $1 \leq j \leq n-1$:
  $A_{l,(n+1)\cdot(i-1)+j} = a(\mathbf{x}_{ij})$
  $A_{l,(n+1)\cdot(i+1)+j} = a(\mathbf{x}_{ij})$
  $A_{l,(n+1)\cdot i+(j-1)} = a(\mathbf{x}_{ij})$
  $A_{l,(n+1)\cdot i+(j+1)} = a(\mathbf{x}_{ij})$

- for the boundary points $\mathbf{x}_{ij}$ with $i = 0$ or $i = m$, or $j = 0$ or $j = n$ and $l = (n+1) \cdot i + j$:
  - for $0 \leq j \leq n$:
    $A_{j,\,(n+1)+j} = 2a(\mathbf{x}_{0j})$
    $A_{(n+1)\cdot m+j,\,(n+1)\cdot(m-1)+j} = 2a(\mathbf{x}_{mj})$

– for $0 \leq i \leq m$:
$A_{(n+1)\cdot i,\,(n+1)\cdot i+1} = 2a(\mathbf{x}_{i0})$
$A_{(n+1)\cdot i+n,\,(n+1)\cdot i+(n-1)} = 2a(\mathbf{x}_{in})$

The matrix A is not tridiagonal as in the one-dimensional case. As its dimension $(m+1)\cdot(n+1)$ is rather big, the system of linear equations cannot be solved exactly, and an approximate solution is sought for.
We have $|A_{ll}| = 1 + 4a(\mathbf{x}_{ij}) = 1 + \sum_{k\neq l} A_{lk} > \sum_{k\neq l} |A_{lk}|$. Therefore, apart from being a sparse matrix, A is also strictly diagonally dominant. Hence the Gauß-Seidel algorithm converges.
We recall that the Gauß-Seidel algorithm is an iterative algorithm to solve a system of linear equations. It starts with an arbitrary initial approximation $x^{(0)}$, and for each iteration step the values $x^{(s+1)}$ are obtained by

$$x_l^{(s+1)} = \frac{1}{A_{ll}} \left(\mathbf{b}_l - \sum_{k<l} A_{lk} x_k^{(s+1)} - \sum_{k>l} A_{lk} x_k^{(s)} \right).$$

Note that the values for an iteration step can be stored at the same place as the values of the previous iteration step, which is a great advantage for high dimensions.
The drawback of the Gauß-Seidel algorithm is its rather slow convergence. It can be speeded up by successive over-relaxation (SOR). This numerical method uses an over-relaxation parameter $\omega \in (0,2)$, and for each iteration step the values $x^{(s+1)}$ of the SOR algorithm are obtained by

$$x_l^{(s+1)} = \omega \cdot \frac{1}{A_{ll}} \left(\mathbf{b}_l - \sum_{k<l} A_{lk} x_k^{(s+1)} - \sum_{k>l} A_{lk} x_k^{(s)} \right) + (1-\omega) \cdot x_l^{(s)}.$$

For $\omega = 1$, one gets back the original Gauß-Seidel method, and for $\omega > 1$ the convergence speeds up with increasing ω. More details about the SOR method can be found in Young [You71] and Saad [Saa03].
We use the SOR method with parameter $\omega = 1.9$ as default to compute the estimator $\hat{f}_a$. As the initial approximation $x^{(0)}$, the zero vector is used.

3.3 Choice of the Smoothing Parameter

This section deals with the choice of the diffusivity in the two-dimensional setting. We start with a presentation of existing diffusion filters. Subsequently we introduce our method, which includes the two-dimensional multiresolution criterion.

3.3.1 Existing Diffusion Filters

A pioneering work in diffusion filters is due to Perona and Malik. They use another version of the two-dimensional inhomogeneous diffusion process, where the diffusivity is located within the Laplace operator:

$$\frac{\partial}{\partial t}u(\mathbf{x},t) = \nabla a(\mathbf{x}) \cdot \nabla u(\mathbf{x},t).$$

This corresponds to the one-dimensional inhomogeneous diffusion process used for the estimator $\hat{f}_a^{\star}$ in Section 2.4. Perona and Malik erroneously call it "anisotropic diffusion" and the version, that we use for the estimator $\hat{f}_a$, "isotropic diffusion" (cf. Perona and Malik [PM87] and [PM90]). Both versions are in fact isotropic.
Perona and Malik choose the diffusivity as a function of the gradient of the data, $a(\mathbf{x}) = D\left(|\nabla u|^2\right)$, in order to sharpen the edges in the image. They propose, among others, to consider the function $D\left(|\nabla u|^2\right) = \frac{1}{1+|\nabla u|^2/\lambda^2}$ with $\lambda > 0$ as a smoothing parameter.
In regions where the data are rather homogeneous, the gradient is small, so the diffusivity should be large. On the other hand, the gradient is large in regions with edges, and there the diffusivity should be small in order to preserve the edge and not to smooth it away.
Weickert [Wei98] has improved the diffusion approach of Perona and Malik by considering anisotropic diffusion, where the diffusivity is regarded as a tensor, not a scalar. This enables to control not only the velocity but also the direction of the diffusion. The diffusion tensor can be chosen so as to avoid smoothing across edges but to enforce smoothing along the edges, resulting in the enhancement of the edges.

3.3.2 The Two-dimensional Multiresolution Criterion

Now we describe our approach where we use a statistical analysis of the residuals to select the local diffusivity. Thus the choice of the diffusivity is also data driven. But the resulting diffusivity $a(\mathbf{x})$ is piecewise constant. Therefore there is almost no difference between the two versions of the inhomogeneous diffusion process, and for reasons of simplicity we choose the first version. Perona and Malik [PM90] have already remarked that in the case of a constant diffusivity the "anisotropic" diffusion reduces to the "isotropic" one.
The advantage of our approach over the existing methods (Perona and Malik, Weickert) is that the smoothing parameter is selected automatically and that one obtains a specific output image. The other methods offer the scale space, in other words a whole set of images in different scales consisting of the solutions at different stopping times τ, out of which the best one has to be chosen manually.

One problem concerning our method is the selection of the local smoothing parameter. This can be overcome by using the multiresolution criterion, as in the one-dimensional case. It provides a decision rule as to whether the residuals $r(\mathbf{x}_{ij}) = y_{ij} - \hat{f}_a(\mathbf{x}_{ij})$ of the estimator $\hat{f}_a$ are to be accepted as Gaussian white noise.
For bivariate data, the multiresolution coefficients are defined as

$$w_P = \frac{1}{\sqrt{|P|}} \sum_{\mathbf{x}_{ij} \in P} r(\mathbf{x}_{ij})$$

for all connected subsets $P \subseteq [0, m] \times [0, n]$. The multiresolution condition requires that

$$|w_P| \leq \sigma_N \sqrt{\delta \log N} \quad \forall\, P \in \mathcal{P},$$

where $N = (m + 1) \cdot (n + 1)$ and $\mathcal{P} \subseteq \mathfrak{P}$ is the set of all connected subsets $P \subseteq [0, m] \times [0, n]$. The value of the constant $\delta > 0$ depends on the constant in the modulus of continuity of the Brownian sheet, which Section 5.2 is concerned with.
We remark that the same variations of the multiresolution criterion as in the one-dimensional case are possible.
For the standard deviation σ of the noise, one uses the robust estimate

$$\sigma_N = \frac{1.481}{2} \ \text{med}(|y_{i,j} - y_{i,j-1} + y_{i-1,j} - y_{i-1,j-1}|,\ 1 \leq i \leq m,\ 1 \leq j \leq n).$$

In analogy with the one-dimensional case, we have $y_{i,j} - y_{i,j-1} + y_{i-1,j} - y_{i-1,j-1} \approx Z_{i,j} - Z_{i,j-1} + Z_{i-1,j} - Z_{i-1,j-1}$, and thus $y_{i,j} - y_{i,j-1} + y_{i-1,j} - y_{i-1,j-1}$ is approximately $\mathcal{N}(0, 4\sigma^2)$-distributed. This yields

$$\text{med}(|y_{i,j} - y_{i,j-1} + y_{i-1,j} - y_{i-1,j-1}|) \approx \frac{2\sigma}{1.481} \quad \forall\, i = 1, \ldots, m \text{ and } j = 1, \ldots, n.$$

Therefore σ_N is an estimate for the standard deviation σ of the noise.

The multiresolution condition is used to select the diffusivity $a(\mathbf{x})$ by an iterative procedure:

1. start with a large constant value $a(\mathbf{x}_{ij}) = \alpha$ for $i = 0, \ldots, m$, $j = 0, \ldots, n$,
2. compute $\hat{f}_a$ and the residuals $r(\mathbf{x}_{ij})$,
3. check the multiresolution constraint $|w_P| \leq \sigma_N \sqrt{\delta \log N}$ in each subset $P \in \mathcal{P}$,

4. reduce the diffusivity $a(\mathbf{x}_{ij})$ by a factor $\lambda < 1$ on all $\mathbf{x}_{ij} \in P$ if the multiresolution constraint is violated in P,

5. repeat steps 2-4 until the multiresolution condition is satisfied in each subset $P \in \mathcal{P}$.

In our implementation we use $\alpha = 500\,\sigma_N/(\max(y_{ij}) - \min(y_{ij}))$ as the initial value for the diffusivity, and $\lambda = 0.8$.

Already in the one-dimensional algorithm, the analysis of the multiresolution condition is restricted to dyadic intervals to make computations feasible. In the two-dimensional case the restriction to specific subsets of the image is even more important, since it is impossible to check every possible region of the image, even for rather small images. Hence a partition $\mathcal{P}$ of the image is needed, which is required to

- include different scales,
- be fine enough to detect also small features,
- allow a fast computation of the multiresolution coefficients,
- facilitate a fast access to all elements of the partition.

We remark that $\mathcal{P}$ is not a partition in the classical sense, which divides a domain into disjoint subdomains. The elements of the partition $\mathcal{P}$ are not disjoint, in fact we have to deal with a union of partitions on different scales. In Section 3.4 two possible partitions $\mathcal{P}$ are introduced. Theorems 5.6 and 5.9 state that for these two partitions all $\delta > 2$ are possible constants in the modulus of continuity, such as for intervals in the one-dimensional case. Again, we use $\delta = 2.3$ as default value for the partition $\mathcal{P}_S$ into dyadic squares, presented in Section 3.4.1.
In Figure 3.3, the resulting estimator $\hat{f}_a$ for the Mexican Hat dataset is shown, where the partition $\mathcal{P}_S$ into dyadic squares is used. The multiscale analysis of the residuals, i.e. the automatic selection of the diffusivity $a(\mathbf{x})$, is carried out on two levels. Further explanation is given in Section 3.4.

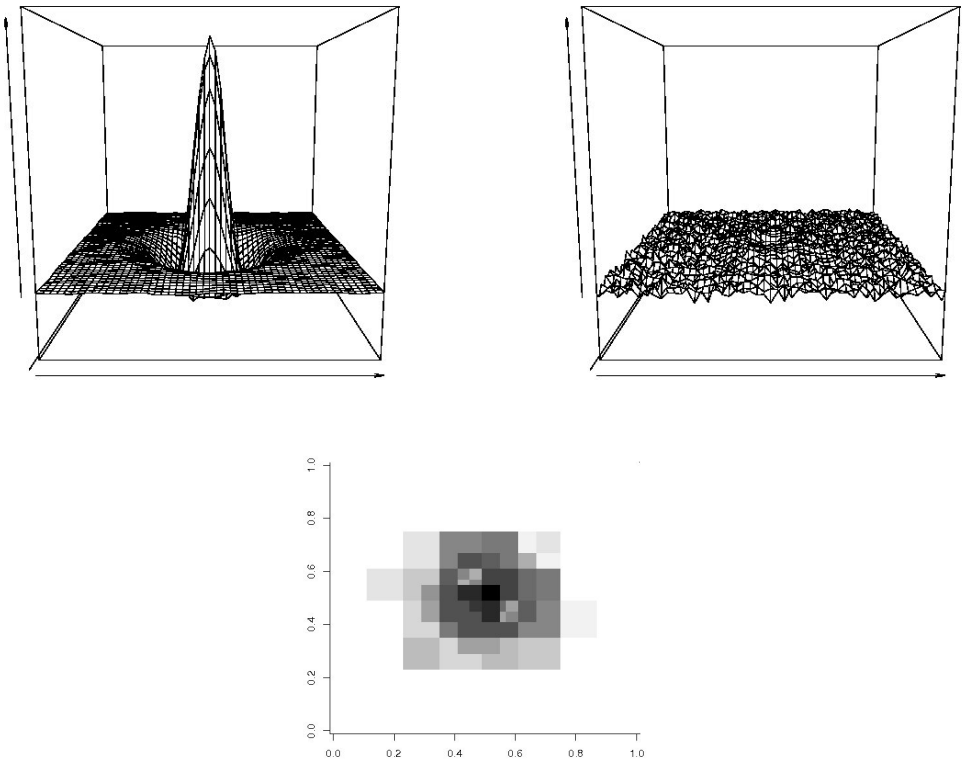


Figure 3.3: Decomposition of the Mexican Hat dataset into the diffusion estimator $\hat{f}_a$ and the residuals
Bottom: the selected local diffusivity $a(\mathbf{x})$ on a logarithmic scale

3.4 Different Partitions $\mathcal{P}$

In this section, two partitions $\mathcal{P}$, which meet the demands formulated above, are introduced: the partition $\mathcal{P}_S$ into dyadic squares, where the image is recursively divided into four parts, and the wedge partition $\mathcal{P}_W$, where each of the dyadic squares is additionally divided into two parts by a straight line.
Subsequently a modest modification of the multiresolution criterion is proposed, which yields better results for data sets with local differences in smoothness. It can be combined with any partition $\mathcal{P}$.

3.4.1 The Partition into Dyadic Squares $\mathcal{P}_S$

The simplest partition of an image is obtained by using the so called dyadic squares, the obvious generalization of dyadic intervals in dimension one. This partition is known from the two-dimensional wavelet transformation. The whole image is divided into four parts, each of which is again divided into four parts and so on until eventually single pixels are attained.
For a quadratic image of $2^k \times 2^k$ pixels, the four sub-elements of the element $P_0 = [0,\, 2^k - 1] \times [0,\, 2^k - 1]$ are given by

- $P_{0,1} = [0,\, 2^{k-1} - 1] \times [0,\, 2^{k-1} - 1]$,
- $P_{0,2} = [2^{k-1},\, 2^k - 1] \times [0,\, 2^{k-1} - 1]$,
- $P_{0,3} = [0,\, 2^{k-1} - 1] \times [2^{k-1},\, 2^k - 1]$,
- $P_{0,4} = [2^{k-1},\, 2^k - 1] \times [2^{k-1},\, 2^k - 1]$.

In this case, each element of $\mathcal{P}_S$ can be written as

$$P = \left[i_1 \cdot 2^{j-1},\, (i_1 + 1) \cdot 2^j - 1\right] \times \left[i_2 \cdot 2^{j-1},\, (i_2 + 1) \cdot 2^j - 1\right],$$

with $j = 1, \ldots, k$ and $i_1, i_2 = 0, \ldots, 2^{k-j}$. The division of an element of $\mathcal{P}_S$ into its four sub-elements is illustrated in Figure 3.4.
The elements of $\mathcal{P}_S$ can be stored in a recursive tree structure. This allows a fast access to all the elements starting with the whole image and recursively passing over to the four sub-elements of an element.
Non-quadratic images and non-dyadic side lengths are also permitted. For a general image of $(m + 1) \times (n + 1)$ pixels, the four sub-elements of the element $P_0 = [0,\, m] \times [0,\, n]$ are given by

- $P_{0,1} = \left[0,\, \lfloor \frac{m}{2} \rfloor\right] \times \left[0,\, \lfloor \frac{n}{2} \rfloor\right]$,
- $P_{0,2} = \left[\lfloor \frac{m}{2} \rfloor + 1,\, m\right] \times \left[0,\, \lfloor \frac{n}{2} \rfloor\right]$,
- $P_{0,3} = \left[0,\, \lfloor \frac{m}{2} \rfloor\right] \times \left[\lfloor \frac{n}{2} \rfloor + 1,\, n\right]$,
- $P_{0,4} = \left[\lfloor \frac{m}{2} \rfloor + 1,\, m\right] \times \left[\lfloor \frac{n}{2} \rfloor + 1,\, n\right]$.

Even though the elements P of the partition $\mathcal{P}_S$ may not be squares, we call them dyadic squares.

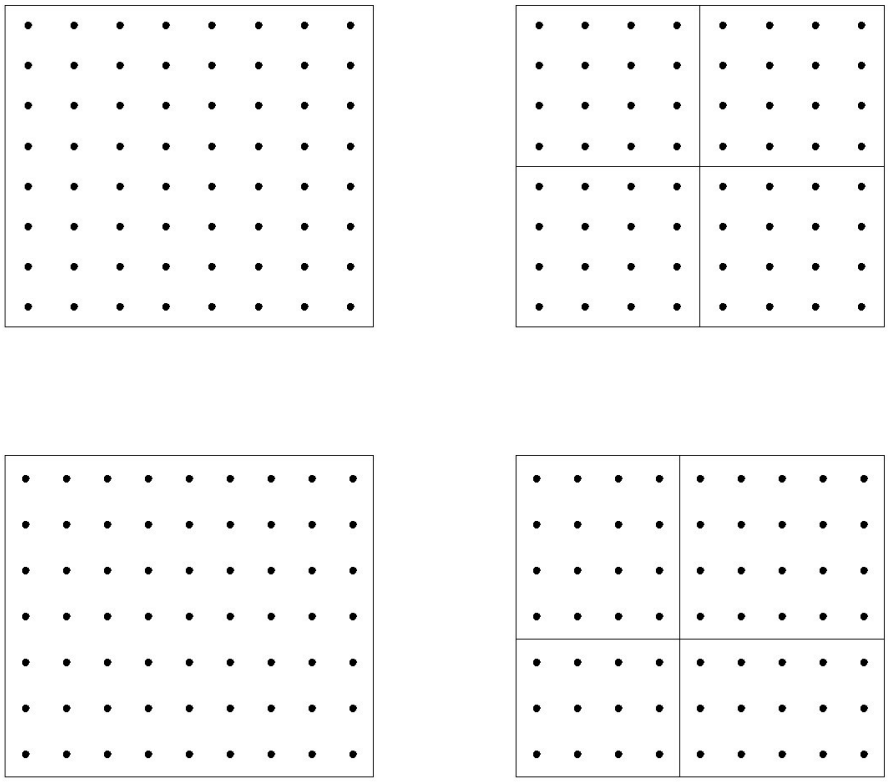


Figure 3.4: Element of $\mathcal{P}_S$ and division into its four sub-elements in the dyadic squares case (top) and in the general case (bottom)

We need to compute the multiresolution coefficients w_P for all elements P of the partition $\mathcal{P}_S$, i.e. to sum up the residuals $r(\mathbf{x}_{ij})$ for $\mathbf{x}_{ij} \in P$ and divide them by $\sqrt{|P|}$. For a fast computation of the sums we consider the matrix of cumulative sums. Let

$$R = (r(\mathbf{x}_{ij}))_{ij}\,, \quad i = 0, \ldots, m,\, j = 0, \ldots, n,$$

be the matrix of the residuals. We define the corresponding matrix of cumulative sums, $\tilde{R}$, as

$$\tilde{R} = \left(\sum_{k=0}^{i} \sum_{l=0}^{j} r(\mathbf{x}_{kl}) \right)_{ij}, \quad i = 0, \ldots, m,\, j = 0, \ldots, n.$$

Let $P = [i_1, i_2] \times [j_1, j_2]$ be an element of the partition $\mathcal{P}_S$. Then we have $|P| = (i_2 - i_1 + 1) \cdot (j_2 - j_1 + 1)$, and the sum of the residuals over $\mathcal{P}$ is ob-

tained by

$$\sum_{\mathbf{x}_{ij} \in P} r(\mathbf{x}_{ij}) = \tilde{R}_{i_2,j_2} - \tilde{R}_{i_1-1,j_2} \, \mathbb{1}_{\{i_1 \geq 1\}} - \tilde{R}_{i_2,j_1-1} \, \mathbb{1}_{\{j_1 \geq 1\}} + \tilde{R}_{i_1-1,j_1-1} \, \mathbb{1}_{\{i_1 \geq 1,\, j_1 \geq 1\}}.$$

This is illustrated in Figure 3.5, where we sum up the residuals over $P = [4, 7] \times [8, 11]$.

Once we have computed the matrix of cumulative sums, the multiresolution coefficients can be computed very fast for all $P \in \mathcal{P}_S$.

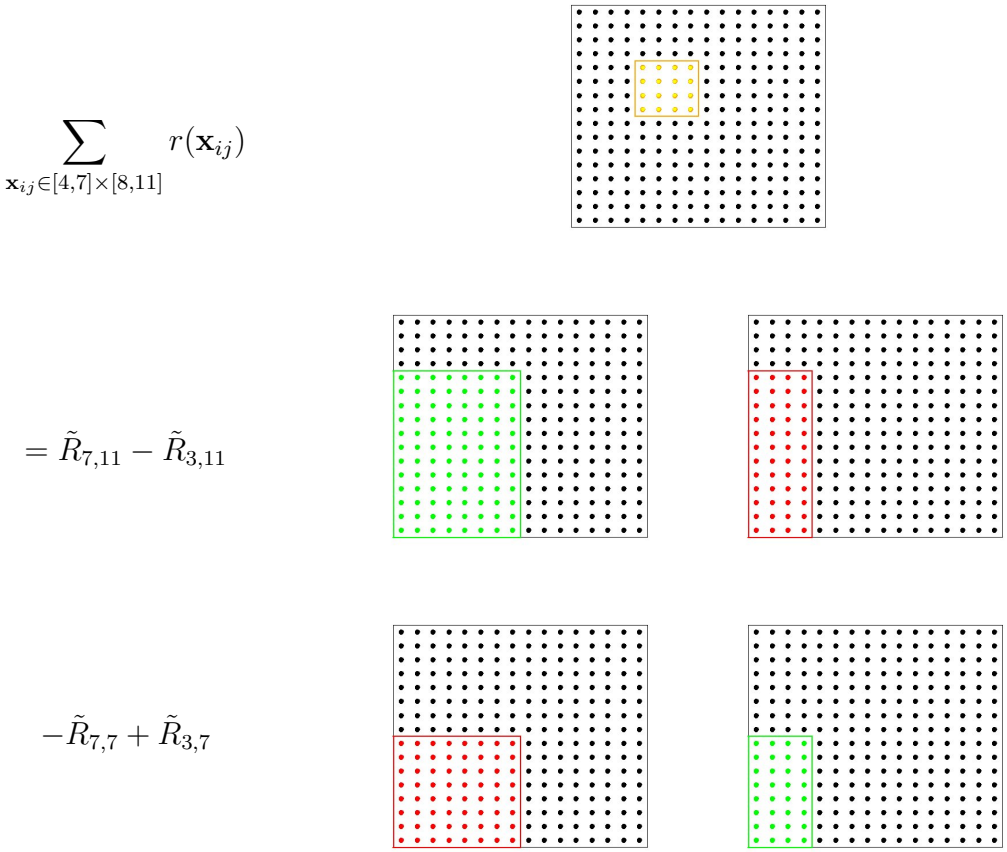


Figure 3.5: Summation of the residuals over a dyadic square $P \in \mathcal{P}_S$

3.4.2 The Wedge Partition $\mathcal{P}_W$

The wedge partition $\mathcal{P}_W$ is a refinement of the partition $\mathcal{P}_S$. It contains the elements of $\mathcal{P}_S$ and additionally the so called wedges, which are obtained by dividing a dyadic square into two parts by a straight line. In the following we present the wedge partition and the algorithm to compute the cumulative sums over wedges. For a more detailed explanation we refer to Friedrich [Fri05] and Friedrich et al. [FDFW07].

First of all, we have to choose a set of angles $\mathcal{A} \subset (-\frac{\pi}{2}, \frac{\pi}{2}]$. For $\alpha \in \mathcal{A}$, the straight line with slope α has to be discretized in order to obtain a digital line. We use the Bresenham algorithm (cf. Bresenham [Bre65]), with a slight change to make it independent of the direction in which the line is drawn, in other words the digital lines for the angles α and $\alpha + \pi$ are the same.

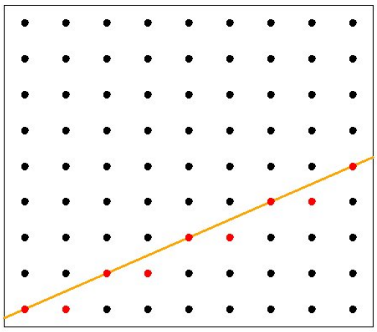


Figure 3.6: Straight line and corresponding digital line

In the following we have to distinguish between horizontal lines, i.e. $\alpha \in (-\frac{\pi}{4}, \frac{\pi}{4}]$, and vertical lines, i.e. $\alpha \in (-\frac{\pi}{2}, \frac{\pi}{2}] \setminus (-\frac{\pi}{4}, \frac{\pi}{4}]$.
From the digital line through the origin with slope α we obtain the values $l_\alpha(i)$, $i \in \mathbb{Z}$, where

- $(i, l_\alpha(i))$ are the pixels on the digital line in the horizontal case,
- $(l_\alpha(i), i)$ are the pixels on the digital line in the vertical case.

Any line is specified by its slope α and the line number n_l, which is given by the pixel $(0, n_l)$ in the case of a horizontal line, respectively the pixel $(n_l, 0)$ in the case of a vertical line. Thus any digital line is specified by the values $l_\alpha(i)$, the line number n_l and the information whether it is a horizontal or a vertical line.

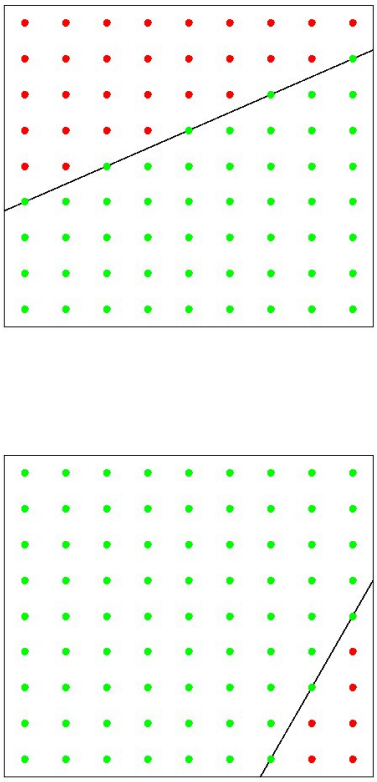


Figure 3.7: Division into lower wedge W_{LO} and upper wedge W_{UP} by the horizontal line with slope $\alpha = \frac{\pi}{6}$ and line number $n_l = 3$, respectively into left wedge W_{LE} and right wegde W_{RI} by the vertical line with slope $\alpha = \frac{\pi}{3}$ and line number $n_l = 6$

A wedge, which is a subset of a dyadic square, is defined as follows: Consider $P \in \mathcal{P}_S$, a digital line given by α and n_l divides P into "lower and upper" or "left and right" parts, which we call wedges.
Now we want to compute the multiresolution coefficients of the wedges. This is done in a similar way as for squares. Firstly we consider horizontal lines, hence we have the lower and upper wedges, which we denote by W_{LO} and W_{UP}, with $W_{LO} \cup W_{UP} = P = [i_1, i_2] \times [j_1, j_2] \in \mathcal{P}_S$.
We compute the matrix of cumulative sums of the residuals, $\tilde{R}_\alpha$, as

$$\tilde{R}_\alpha = \left(\sum_{k=0}^{i} \sum_{l=0}^{l_\alpha(k)+n_l(\alpha,i,j)} r(\mathbf{x}_{kl}) \right)_{ij}, \quad i = 0, \ldots, m,\ j = 0, \ldots, n,$$

where $n_l(\alpha, i, j)$ is the line number of the digital line with slope α going through

the pixel (i, j). That means, for each pixel (i, j) a line with slope α, which goes through the pixel is drawn, and the values of the residuals left of i and up to this line are summed up.
We consider the case where $l_\alpha(i_2) + n_l \leq j_2$ (which means that the line intersects the dyadic square on the right side of P), and we start with the lower wedge W_{LO}. The sum of the residuals over W_{LO} can be computed as follows,

$$\sum_{\mathbf{x}_{ij} \in W_{LO}} r(\mathbf{x}_{ij}) = \tilde{R}_\alpha(i_2, l_\alpha(i_2) + n_l) - \tilde{R}_\alpha(i_1 - 1, l_\alpha(i_1 - 1) + n_l)\, \mathbb{1}_{\{i_1 \geq 1\}}$$

$$-\tilde{R}(i_2, j_1 - 1), \mathbb{1}_{\{j_1 \geq 1\}} + \tilde{R}(i_1 - 1, j_1 - 1)\, \mathbb{1}_{\{i_1 \geq 1,\, j_1 \geq 1\}}.$$

An example is given in Figure 3.8. It shows the lower wedge W_{LO}, which emerges by dividing the dyadic square $P = [4, 7] \times [8, 11]$ by the line with slope $\alpha = \frac{\pi}{6}$ and line number $n_l = 7$ and illustrates the summation of the residuals over W_{LO}.
For the multiresolution coefficient, one additionally needs the size $|W_{LO}|$ of the wedge. Therefore we compute the matrix $\tilde{S}_\alpha$, whose entries are the cumulative sums of the matrix $S = (1)_{ij}$,

$$\tilde{S}_\alpha = \left(\sum_{k=0}^{i} \sum_{l=0}^{l_\alpha(k)+n_l(\alpha,i,j)} 1 \right)_{ij}, \quad i = 0, \ldots, m,\ j = 0, \ldots, n.$$

Hence one obtains

$$|W_{LO}| = \sum_{\mathbf{x}_{ij} \in W_{LO}} 1 = \tilde{S}_\alpha(i_2, l_\alpha(i_2)+n_l) - \tilde{S}_\alpha(i_1-1, l_\alpha(i_1-1)+n_l)\, \mathbb{1}_{\{i_1 \geq 1\}} - (i_2 - i_1 + 1) \cdot j_1,$$

and the multiresolution coefficient

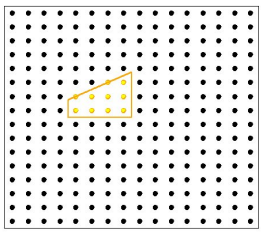
$$w_{W_{LO}} = \frac{1}{\sqrt{|W_{LO}|}} \sum_{\mathbf{x}_{ij} \in W_{LO}} r(\mathbf{x}_{ij})$$

can be computed.
Now let $l_\alpha(i_2) + n_l > j_2$. In this case, the sum of the residuals over W_{LO} is

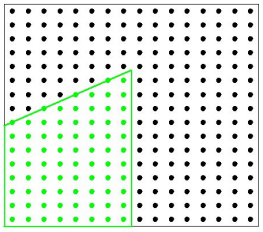
$$\sum_{\mathbf{x}_{ij} \in W_{LO}} r(\mathbf{x}_{ij}) = \tilde{R}_\alpha(i(\alpha, n_l), l_\alpha(i(\alpha, n_l)) + n_l) - \tilde{R}_\alpha(i_1 - 1, l_\alpha(i_1 - 1) + n_l)\, \mathbb{1}_{\{i_1 \geq 1\}}$$

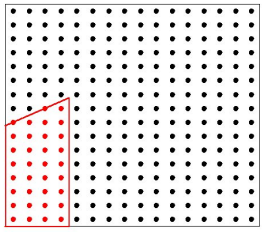
$$-\tilde{R}(i(\alpha, n_l), j_1 - 1)\, \mathbb{1}_{\{j_1 \geq 1\}} + \tilde{R}(i_1 - 1, j_1 - 1)\, \mathbb{1}_{\{i_1 \geq 1,\, j_1 \geq 1\}}$$
$$+\tilde{R}(i_2, j_2) - \tilde{R}(i(\alpha, n_l), j_2)\, \mathbb{1}_{\{i(\alpha, n_l) \geq 0\}}$$
$$-\tilde{R}(i_2, j_1 - 1)\, \mathbb{1}_{\{j_1 \geq 1\}} + \tilde{R}(i(\alpha, n_l), j_1 - 1)\, \mathbb{1}_{\{i(\alpha, n_l) \geq 0,\, j_1 \geq 1\}},$$

$$\sum_{\mathbf{x}_{ij} \in W_{LO}} r(\mathbf{x}_{ij})$$

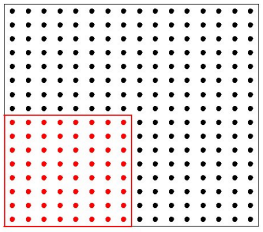


$$= \tilde{R}_\alpha(7,10) - \tilde{R}_\alpha(3,8)$$





$$-\tilde{R}(7,7) + \tilde{R}(3,7)$$



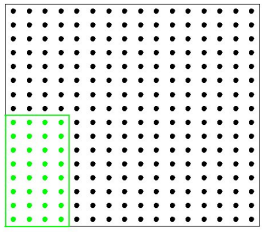


Figure 3.8: Summation of the residuals over a wedge $W \in \mathcal{P}_W$

where $i(\alpha, n_l)$ is the smallest integer i with $l_\alpha(i) + n_l = j_2$. That means that the pixel $(i(\alpha, n_l), j_2)$ is the intersection of the line and the dyadic square. This is illustrated in Figure 3.9 with $P = [0,7] \times [0,7]$ being the whole image, $\alpha = \frac{\pi}{6}$, $n_l = 6$ and hence $i(\alpha, n_l) = 4$.
The size of the lower wedge is given by

$$|W_{LO}| = \sum_{\mathbf{x}_{ij} \in W_{LO}} 1 = \tilde{S}_\alpha(i(\alpha, n_l), l_\alpha(i(\alpha, n_l)) + n_l) - \tilde{S}_\alpha(i_1 - 1, l_\alpha(i_1 - 1) + n_l)\, \mathbb{1}_{\{i_1 \geq 1\}}$$

$$-(i(\alpha,n_l)+1)\cdot j_1 + i_1 \cdot j_1 + (i_2+1)\cdot(j_2+1) - (i(\alpha,n_l)+1)\cdot(j_2+1) - (i_2+1)\cdot j_1 + (i(\alpha,n_l)+1)\cdot j_1$$

$$= \tilde{S}_\alpha(i(\alpha, n_l), l_\alpha(i(\alpha, n_l)) + n_l) - \tilde{S}_\alpha(i_1 - 1, l_\alpha(i_1 - 1) + n_l)\, \mathbb{1}_{\{i_1 \geq 1\}}$$

$$-(i_2 - i_1 + 1) \cdot j_1 + (i_2 - i(\alpha, n_l)) \cdot (j_2 + 1).$$

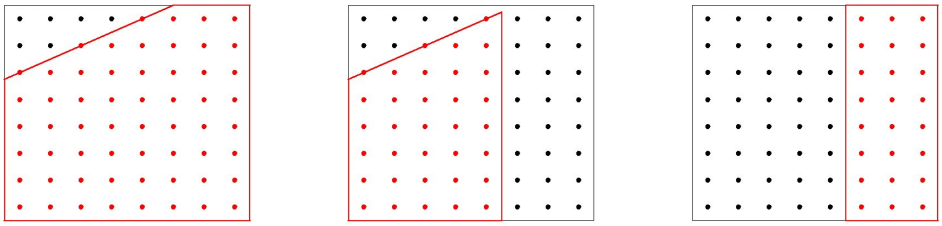


Figure 3.9: The sum of the residuals over the wedge is obtained by adding the sum over the trapezium to the sum over the rectangle.

Once one has the values for the lower wedge W_{LO}, the correponding values for the upper wedge W_{UP} are simply obtained by

$$\sum_{\mathbf{x}_{ij}\in W_{UP}} r(\mathbf{x}_{ij}) = \sum_{\mathbf{x}_{ij}\in P} r(\mathbf{x}_{ij}) - \sum_{\mathbf{x}_{ij}\in W_{LO}} r(\mathbf{x}_{ij}) \quad \text{and}$$

$$|W_{UP}| = |P| - |W_{LO}| = (i_2 - i_1 + 1)\cdot(j_2 - j_1 + 1) - |W_{LO}|.$$

This is illustrated in Figure 3.10.

The computations for vertical lines and the corresponding wedges W_{LE} and W_{RI} are analogous, the roles of the x- and y-axis are interchanged.
The matrices of cumulative sums $\tilde{R}_\alpha$ and $\tilde{S}_\alpha$ are defined as follows:

$$\tilde{R}_\alpha = \left(\sum_{l=0}^{j} \sum_{k=0}^{l_\alpha(l)+n_l(\alpha,i,j)} r(\mathbf{x}_{kl}) \right)_{ij}, \quad i = 0,\dots,m,\ j = 0,\dots,n \quad \text{and}$$

$$\tilde{S}_\alpha = \left(\sum_{l=0}^{j} \sum_{k=0}^{l_\alpha(l)+n_l(\alpha,i,j)} 1 \right)_{ij}, \quad i = 0,\dots,m,\ j = 0,\dots,n.$$

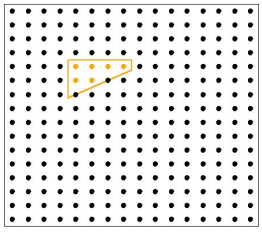
For $l_\alpha(j_2) + n_l \le i_2$, one obtains

$$\sum_{\mathbf{x}_{ij}\in W_{LE}} r(\mathbf{x}_{ij}) = \tilde{R}_\alpha(l_\alpha(j_2)+n_l, j_2) - \tilde{R}_\alpha(l_\alpha(j_1-1)+n_l, j_1-1)\,\mathbb{1}_{\{j_1\ge 1\}}$$

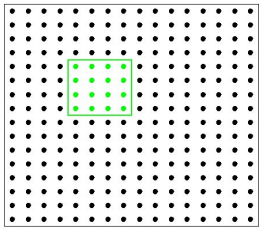
$$-\tilde{R}(i_1-1, j_2)\,\mathbb{1}_{\{i_1\ge 1\}} + \tilde{R}(i_1-1, j_1-1)\,\mathbb{1}_{\{i_1\ge 1,\, j_1\ge 1\}} \quad \text{and}$$

$$|W_{LE}| = \sum_{\mathbf{x}_{ij}\in W_{LE}} 1 = \tilde{S}_\alpha(l_\alpha(j_2)+n_l, j_2) - \tilde{S}_\alpha(l_\alpha(j_1-1)+n_l, j_1-1)\,\mathbb{1}_{\{j_1\ge 1\}} - (j_2-j_1+1)\cdot i_1,$$

$$\sum_{\mathbf{x}_{ij}\in W_{LO}} r(\mathbf{x}_{ij})$$



$$= \sum_{\mathbf{x}_{ij}\in P} r(\mathbf{x}_{ij}) - \sum_{\mathbf{x}_{ij}\in W_{UP}} r(\mathbf{x}_{ij}$$



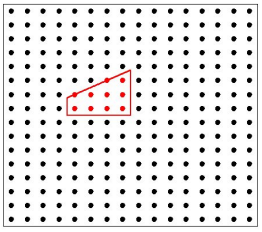


Figure 3.10: The upper wedge W_{UP}, which arises from the division of the dyadic square $P = [4, 7] \times [8, 11]$ by the line with slope $\alpha = \frac{\pi}{6}$ and line number $n_l = 7$.

and for $l_\alpha(j_2) + n_l > i_2$, one obtains with $j(\alpha, n_l)$ being the smallest integer j satisfying $l_\alpha(j) + n_l = i_2$

$$\sum_{\mathbf{x}_{ij}\in W_{LE}} r(\mathbf{x}_{ij}) = \tilde{R}_\alpha(l_\alpha(j(\alpha, n_l)) + n_l, l_\alpha(j(\alpha, n_l))) - \tilde{R}_\alpha(l_\alpha(j_1 - 1) + n_l, j_1 - 1)\, \mathbb{1}_{\{j_1\geq 1\}}$$

$$-\tilde{R}(i_1 - 1, j(\alpha, n_l))\, \mathbb{1}_{\{i_1\geq 1\}} + \tilde{R}(i_1 - 1, j_1 - 1)\, \mathbb{1}_{\{i_1\geq 1, j_1\geq 1\}}$$

$$+\tilde{R}(i_2, j_2) - \tilde{R}(i_2, j(\alpha, n_l))\, \mathbb{1}_{\{j(\alpha,n_l)\geq 0\}}$$

$$-\tilde{R}(i_1 - 1, j_2)\, \mathbb{1}_{\{i_1\geq 1\}} + \tilde{R}(i_1 - 1, j(\alpha, n_l))\, \mathbb{1}_{\{i_1\geq 1, j(\alpha,n_l)\geq 0\}} \quad \text{and}$$

$$|W_{LE}| = \sum_{\mathbf{x}_{ij}\in W_{LE}} 1 = \tilde{S}_\alpha(l_\alpha(j(\alpha, n_l)) + n_l, j(\alpha, n_l)) - \tilde{S}_\alpha(l_\alpha(j_1 - 1) + n_l, j_1 - 1)\, \mathbb{1}_{\{j_1\geq 1\}}$$

$$-i_1 \cdot (j_2 - j_1 + 1) + (i_2 + 1) \cdot (j_2 - j(\alpha, n_l)).$$

The values for the right wedge W_{RI} are then obtained by

$$\sum_{\mathbf{x}_{ij}\in W_{RI}} r(\mathbf{x}_{ij}) = \sum_{\mathbf{x}_{ij}\in P} r(\mathbf{x}_{ij}) - \sum_{\mathbf{x}_{ij}\in W_{LE}} r(\mathbf{x}_{ij}) \quad \text{and}$$

$$|W_{RI}| = |P| - |W_{LE}| = (i_2 - i_1 + 1) \cdot (j_2 - j_1 + 1) - |W_{LE}|.$$

We write $W_1(P, \alpha, n_l)$ for the lower wedge W_{LO}, which arises from the division of the dyadic square P by the horizontal line with slope α and line number n_l, respectively for the left wedge W_{LE} in the case of a vertical line. Likewise we write $W_2(P, \alpha, n_l)$ for the corresponding upper wedge W_{UP}, respectively for the right wedge W_{RI}. Then the partition $\mathcal{P}_W$ can be written as

$$\mathcal{P}_W = \mathcal{P}_W(\mathcal{A}) = \{W_1(P, \alpha, n_l),\ W_2(P, \alpha, n_l),\ P \in \mathcal{P}_S,\ \alpha \in \mathcal{A},\ 0 \leq n_l \leq \max(m, n)\}.$$

In order to check the multiresolution criterion for each wedge of the partition $\mathcal{P}_W$, one starts with a specific angle $\alpha \in \mathcal{A}$ and computes the matrices $\tilde{R}_\alpha$ and $\tilde{S}_\alpha$. Then the tree of dyadic squares is traversed. For each square $P \in \mathcal{P}_S$, we consider the wedges, which result from the division of the square by the lines with slope α and different line numbers. The multiresolution coefficient is easily computed for all these wedges by means of $\tilde{R}_\alpha$ and $\tilde{S}_\alpha$. This procedure is repeated for each angle $\alpha \in \mathcal{A}$.
The partition $\mathcal{P}_W$ fulfills the postulated requirements because of the tree structure of $\mathcal{P}_S$ and the use of the matrices $\tilde{R}_\alpha$ and $\tilde{S}_\alpha$.

3.4.3 Multiscale Analysis on Two Levels

In this section we present a slight modification of the multiresolution criterion. In the multiscale analysis, the residuals are analysed in each iteration step to decide whether they include only the noise or also parts of the signal. If the absolute value of the multiresolution coefficient, $|w_P|$, passes a threshold, the diffusivity in P is decreased. It is possible though, that the approximation is very poor just in a small subset of P and is good in the remaining pixels of P. In this case, one preferably decreases the diffusivity only in the small subset, which may permit a large diffusivity in the rest of P, resulting in a smoother approximation.
In order to achieve this, the multiscale analysis of the residuals is carried out on two levels. Therefore the partition $\mathcal{P}$ is split into two parts: $\mathcal{P}^{small}$ and $\mathcal{P}^{large}$, where $\mathcal{P}^{small}$ contains all the elements of $\mathcal{P}$ up to a certain size and $\mathcal{P}^{large}$ contains the remaining elements of $\mathcal{P}$.
On the first level, the multiresolution criterion is checked only for $P \in \mathcal{P}^{small}$, and the diffusivity $a(\mathbf{x})$ is adapted so that the multiresolution condition is satisfied for each $P \in \mathcal{P}^{small}$. After this is accomplished, the remaining subsets $P \in \mathcal{P}^{large}$ are analysed, and if necessary the diffusivity $a(\mathbf{x})$ is further decreased.
The multiscale analysis on two levels could, depending on the data, result in a larger diffusivity in regions with little variability. The subsets, where the diffusivity $a(\mathbf{x})$ is decreased, are better focused on the regions where the data have a large

variability.
For the partitions $\mathcal{P}_S$ and $\mathcal{P}_W$, the multiscale analysis on two levels is easily implemented. $\mathcal{P}^{small}$ includes the dyadic squares whose side lengths are smaller than a specific value. On the first level, the tree of dyadic squares is traversed starting with all squares of the specified side lengths. On the second level, the tree is traversed starting with the whole image up to the dyadic squares of the specified side lengths. In our implementation, $\mathcal{P}^{small}$ includes by default the dyadic squares whose smaller side length is shorter than 8.

3.5 Numerical Results

In this section, we present numerical results for the two-dimensional diffusion estimator $\hat{f}_a$. As an example we consider the Peak dataset (cf. Appendix A.2.2).

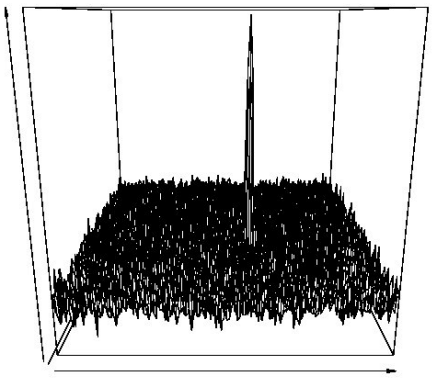


Figure 3.11: Peak dataset

We start with a comparison of the diffusion estimator using the different partitions and the different types of multiscale analysis discussed in Section 3.4. Subsequently we compare the two-dimensional diffusion estimator to other two-dimensional smoothing methods, namely the adaptive weights smoothing and the wavelet thresholding.

3.5.1 Different Versions of the Two-dimensional Diffusion Estimator $\hat{f}_a$

Figures 3.12, 3.13 and 3.14 display the results of the diffusion estimator $\hat{f}_a$, where different versions of the multiscale analysis are used for the automatic selection of

the diffusivity $a(\mathbf{x})$. Each figure shows the estimator $\hat{f}_a$ and the selected diffusivity $a(\mathbf{x})$, which is plotted on a logarithmic scale. The differences in the performance are better visible in the cuts through the line number 150, where the peak is located.

Firstly we compare the results using the different partitions, namely the partition $\mathcal{P}_S$ into dyadic squares and the wedge partition $\mathcal{P}_W$. The partition $\mathcal{P}_S$ into dyadic squares yields at the same time a smoother result and a better approximation of the height of the peak, moreover it needs less computing time. The interpretation is analogous to the interpretation of the difference between the one-dimensional partitions into dyadic intervals and into all intervals (cf. Section 2.6). The wedge partition $\mathcal{P}_W$ contains more regions of the image, therefore a larger value of δ has to be chosen. Here we have used $\delta = 5.5$, which yields at the same time a smooth result and a reasonable approximation of the peak.
Using the partition $\mathcal{P}_S$, we want to compare the results for the multiscale analysis on two levels to the results using the normal multiscale analysis. The first approximations in the iterative procedure, i.e. the approximations with a large diffusivity are so far away from the data at the peak, that the multiresolution condition is not satisfied on the whole image initially, and later on the quarter of the image, where the peak is situated. Therefore the diffusivity $a(\mathbf{x})$ is decreased in those regions, if the normal multiresolution criterion is applied. But note that it is sufficient to decrease $a(\mathbf{x})$ at the peak, and a large diffusivity in the remaining pixels leads to an approximation which satisfies the multiresolution criterion on the whole partition $\mathcal{P}$. Therefore the multiscale analysis on two levels yields smoother results and a better approximation of small features than the normal multiscale analysis.

We have seen that the partition $\mathcal{P}_S$ into dyadic squares is to be preferred to the wedge partition $\mathcal{P}_W$, and that the multiscale analysis on two levels achieves better results than the normal multiscale analysis. Hence in the following, we will use the version of the diffusion estimator $\hat{f}_a$, where the diffusivity is selected by a multiscale analysis on two levels of the partition $\mathcal{P}_S$.

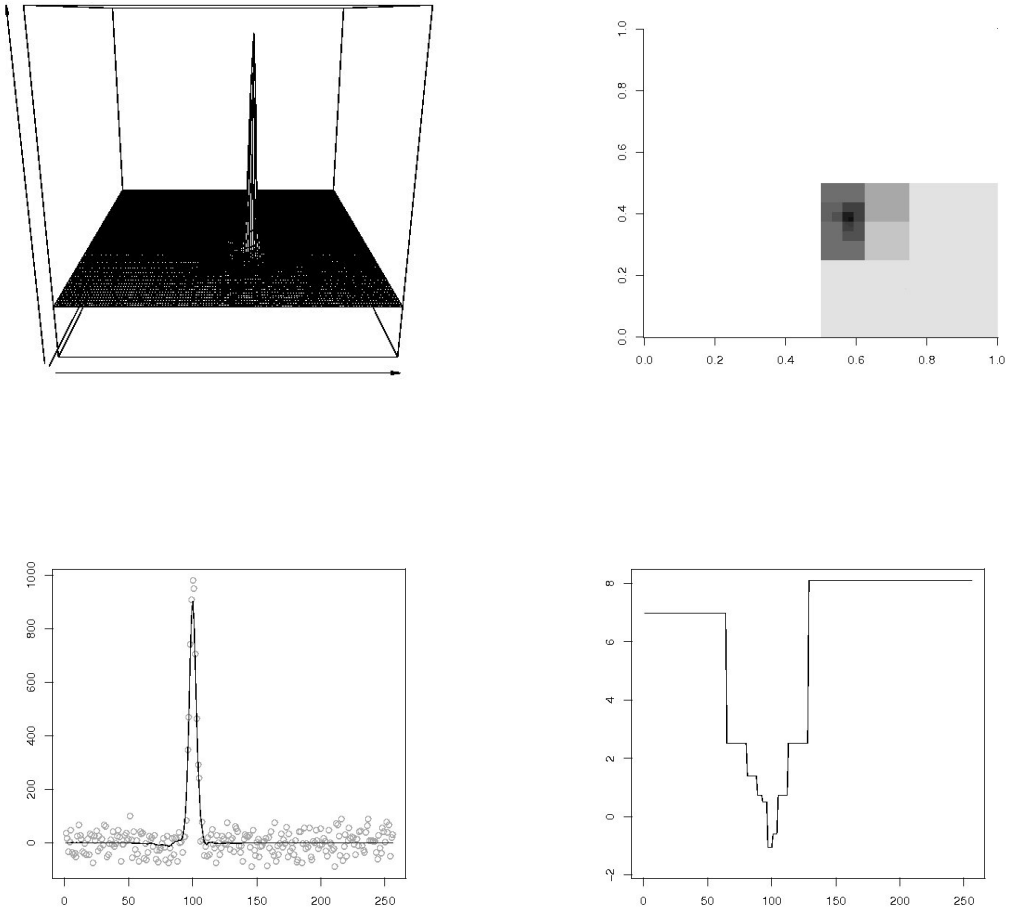


Figure 3.12: Diffusion estimator $\hat{f}_a$ and the diffusivity $a(\mathbf{x})$, selected by a multiscale analysis of the residuals using the partition $\mathcal{P}_S$ into dyadic squares

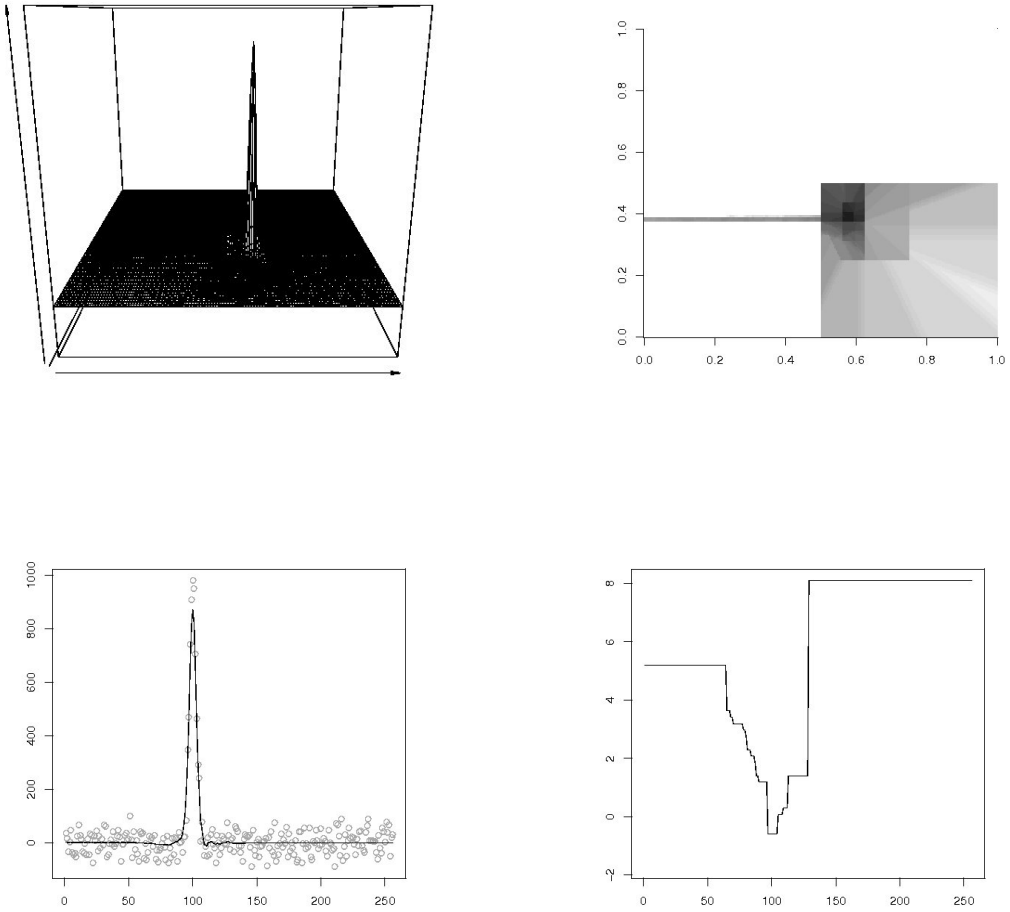


Figure 3.13: Diffusion estimator $\hat{f}_a$ and the diffusivity $a(\mathbf{x})$, selected by a multiscale analysis of the residuals using the wedge partition $\mathcal{P}_W$

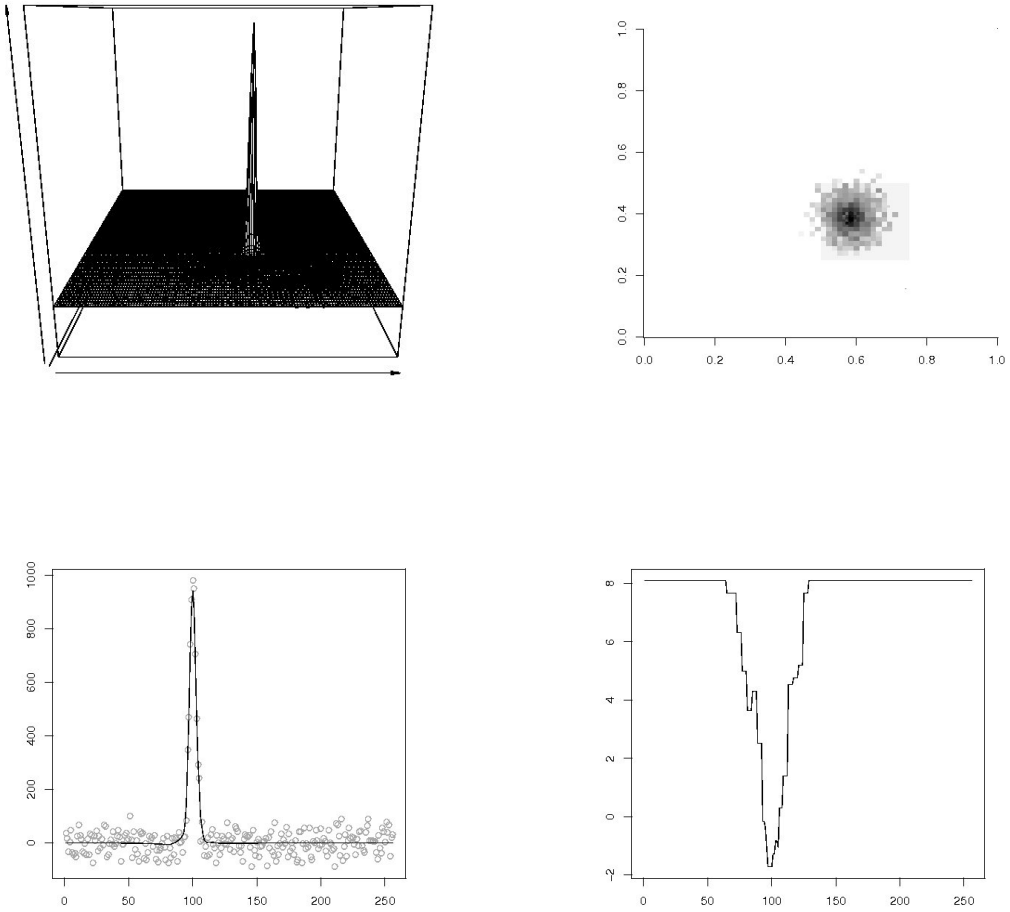


Figure 3.14: Diffusion estimator $\hat{f}_a$ and the diffusivity $a(\mathbf{x})$, selected by a multiscale analysis of the residuals on two levels using the partition $\mathcal{P}_S$ into dyadic squares

3.5.2 Comparison of the Two-dimensional Diffusion Estimator to Other Smoothing Methods

Now we compare the diffusion estimator $\hat{f}_a$ to the results of the two-dimensional adaptive weights smoothing method and the hard wavelet thresholding. These are computed by the functions `aws` (R–package `aws` [Pol06]) and the functions `imwd`, `threshold` and `imwr` (R–package `wavethresh` [NKM06], [Nas93] and [NS94]). We use two different wavelet families: the Daubechies extremal phase wavelets, which

yield rather smooth solutions, and the Haar wavelets, which perform very well with discontinuous signals. For the wavelet thresholding, the MAD is used to estimate the standard deviation σ of the noise.
The inhomogeneous diffusion estimator $\hat{f}_a$ is computed by the function `diffest2d`, which is given in Appendix B.2. Here the diffusivity a is selected by a multiscale analysis of the residuals on two levels using the partition $\mathcal{P}_S$ into dyadic squares. We have seen above that this version of $\hat{f}_a$ is to be favored.

The peak signal-to-noise ratio, PSNR, is used as a measure of the quality of reconstruction in image processing (cf. Wang et al. [WBSS04] and Netravali and Haskell [NH95]). For a signal A and a reconstruction B with mean squared error MSE, the PSNR is defined as

$$\text{PSNR} = 10 \log_{10} \frac{(\max A)^2}{\text{MSE}}.$$

A high value is desirable for the PSNR. Its median from 100 simulations is recorded in Table 3.1 for the different estimators. The diffusion estimator $\hat{f}_a$ attains the best results in terms of the PSNR, the values using different partitions and multiscale analyses for the selection of the diffusivity a are comparable.

Diffusion estimator $\hat{f}_a$ with $\mathcal{P}_S$	51.86
Diffusion estimator $\hat{f}_a$ with $\mathcal{P}_S$ on two levels	52.72
Diffusion estimator $\hat{f}_a$ with $\mathcal{P}_W$	52.03
Adaptive weights smoothing	47.29
Daubechies wavelets	45.63
Haar wavelets	47.77

Table 3.1: PSNR for the Peak test signal and different estimators

As we have already commented in Section 2.7 concerning the MSE in the one-dimensional setting, a single value is not adequate to assess the performance of an estimator. Therefore we place the emphasis on the plots, the PSNR is only mentioned for the sake of completeness.
Figures 3.15, 3.16, 3.17 and 3.18 show the resulting estimators for the Peak dataset. Again the cuts through the line, where the peak is located, are plotted for a better visibility of the performance.

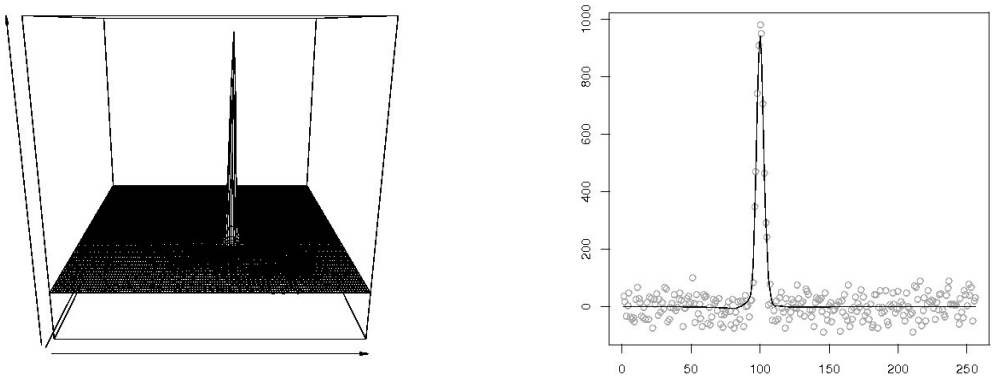


Figure 3.15: The diffusion estimator $\hat{f}_a$ for the Peak dataset

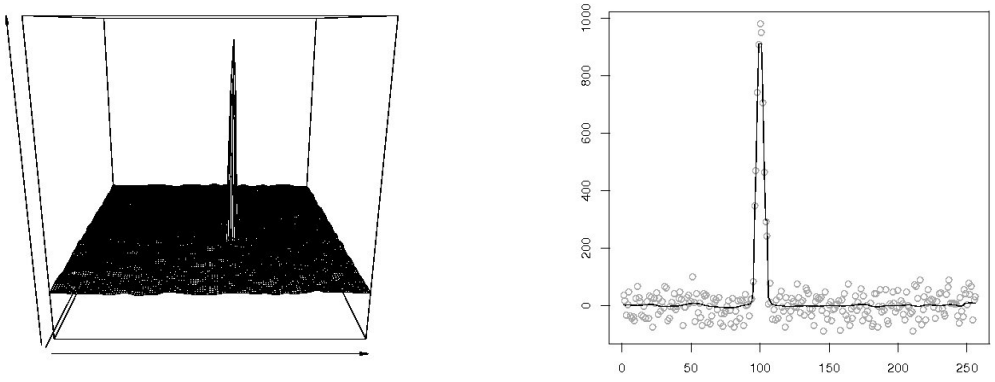


Figure 3.16: The adaptive weights smoothing estimator for the Peak dataset

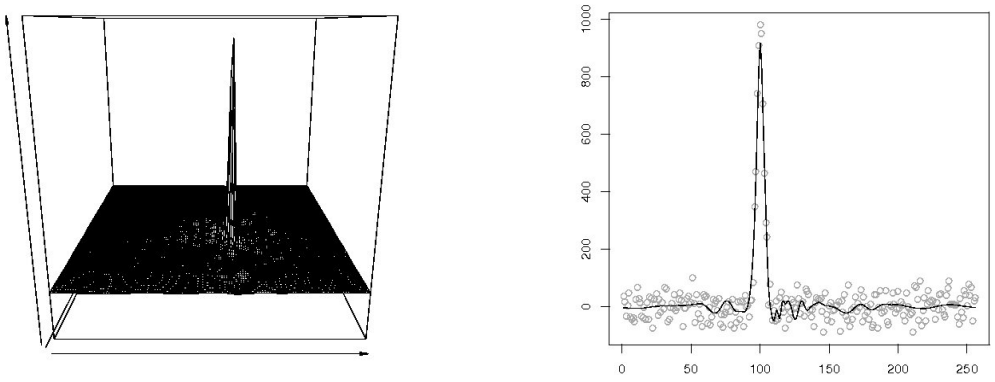


Figure 3.17: Hard wavelet thresholding using Daubechies wavelets with extremal phase for the Peak dataset

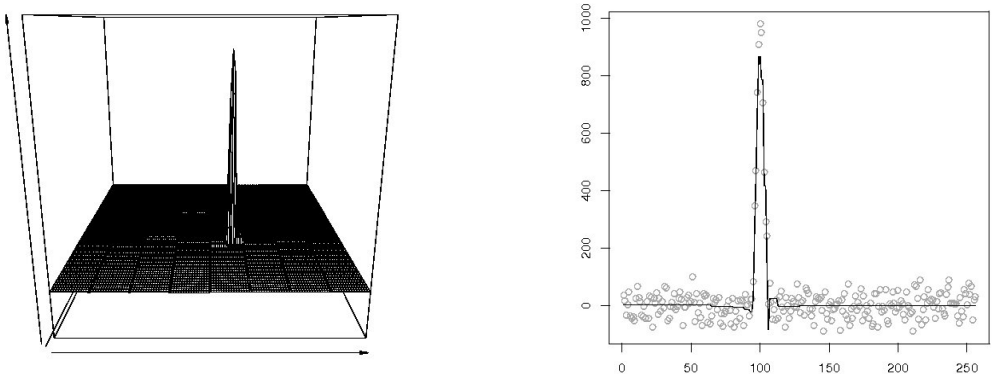


Figure 3.18: Hard wavelet thresholding using Haar wavelets for the Peak dataset

The best results in terms of smoothness and approximation of the peak are achieved by the diffusion estimator. The adaptive weights smoothing also yields good results. However the approximation is less smooth outside the peak. Another disadvantage using the adaptive weights smoothing for smooth datasets is the piecewise constancy of the approximation. The wavelet thresholding causes artefacts like in the one-dimensional setting, the resulting approximations are not satisfactory.

3.6 An Application from Quantum Information Science

So far we have only considered artificial two-dimensional datasets. Now we want to apply our method to real data. In the following we examine the Quantum Computing dataset, made available by courtesy of Emre Togan from the Department of Applied Physics at Harvard University.

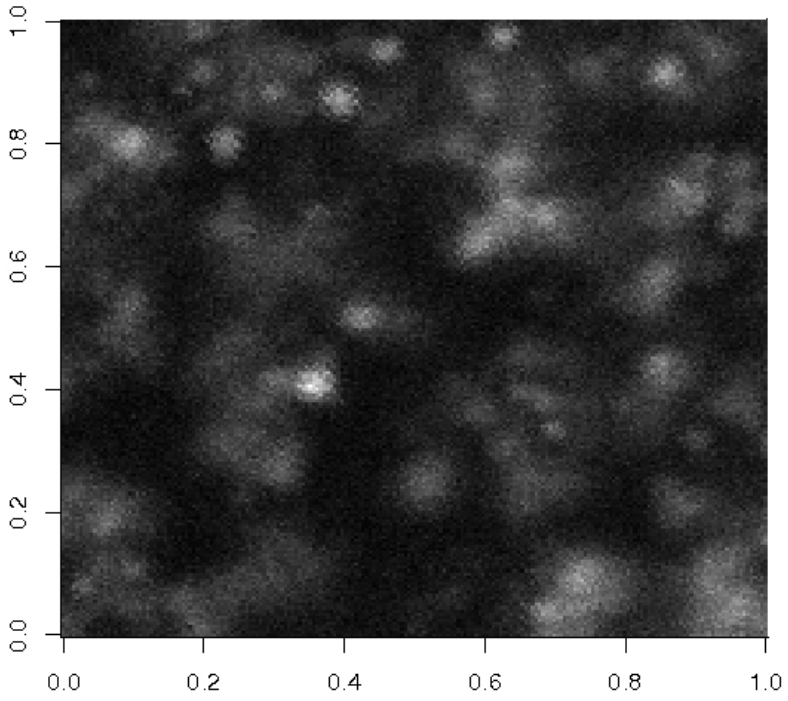


Figure 3.19: The Quantum Computing dataset

The data show the photoluminescence in a diamond sample. The aim of the research group is to point out the peaks of high photoluminiscence since they mark the so called nitrogene-vacancy (NV) centres in the diamond. An NV centre consists of a nitrogene atom located next to a vacant site in the lattice. Knowledge of the location of NV centres enables focusing the microscope on a single NV centre to make further experiments and gain more information about it.
The interest in NV centres in diamond is due to their applicability in quantum information science which is an emerging research field. In a quantum computer the information is stored in qubits (quantum bits). Of the many systems that are

studied as qubits, NV centres in diamond have the advantage of being a solid state system where the spin of single NVs can be manipulated at room temperature. Therefore a quantum computer working at room temperature based on the NV centre of diamond is imaginable. We refer to [DCJ+07], [CDT+06], [EMKA05] and [Ken05] for more detailed information about the research in this field.
Figure 3.20 shows the result of the diffusion estimator with the default settings. The multiscale analysis is carried out on two levels for the partition $\mathcal{P}_S$ into dyadic squares. Apart from the processed image, we show a vertical cut through the image at the line number 71, which contains the highest peak, and a horizontal cut at the line number 196.
With real datasets, usually the contained noise is not exactly Gaussian but might include some structure itself. Therefore the result of our diffusion estimator at some places is not as smooth as we would like it to be. Therefore we have altered the default settings. Figure 3.21 shows the results with $\delta = 15$ and initial diffusivity $a = 200 \cdot \sigma_N$. A higher value for δ means more tolerance in what is accepted to be the noise.
The physicists were satisfied with both versions. They want to find certain patterns in the data, namely the peaks of high photoluminiscence, and do so by comparing the image to other images taken before. The absence of noise speeds up this procedure considerably, both using the processed image with the default settings and the one with the altered settings. In both processed images, the important information is kept. Additionally in some places which appear rather blurry in the raw data, some realistic patterns become visible in the processed data. These patterns are also included in the raw data, but it is more difficult to find them without the indication from the processed data.

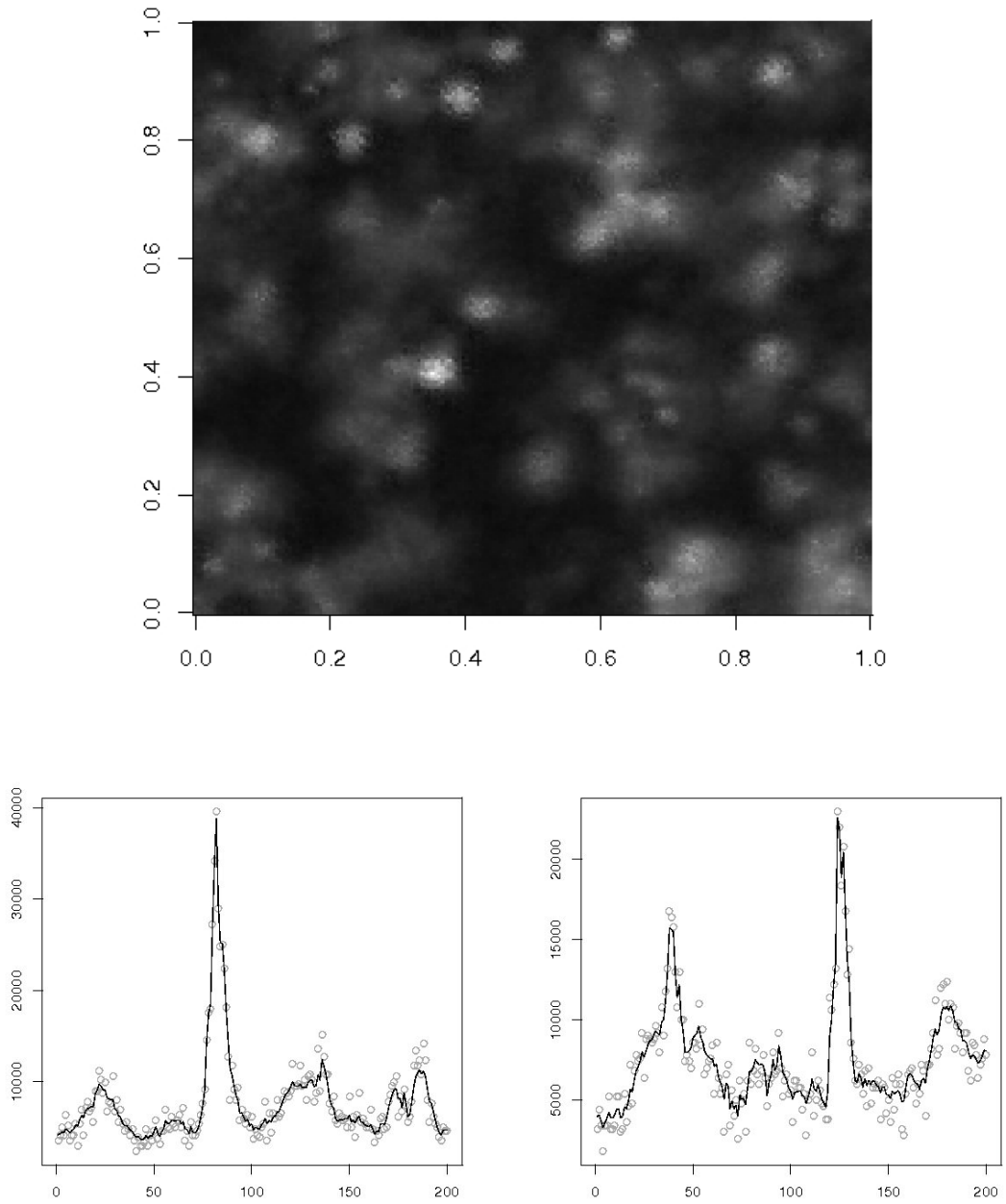


Figure 3.20: Diffusion estimator $\hat{f}_a$ for the Quantum Computing dataset with default settings
Bottom: vertical and horizontal cuts

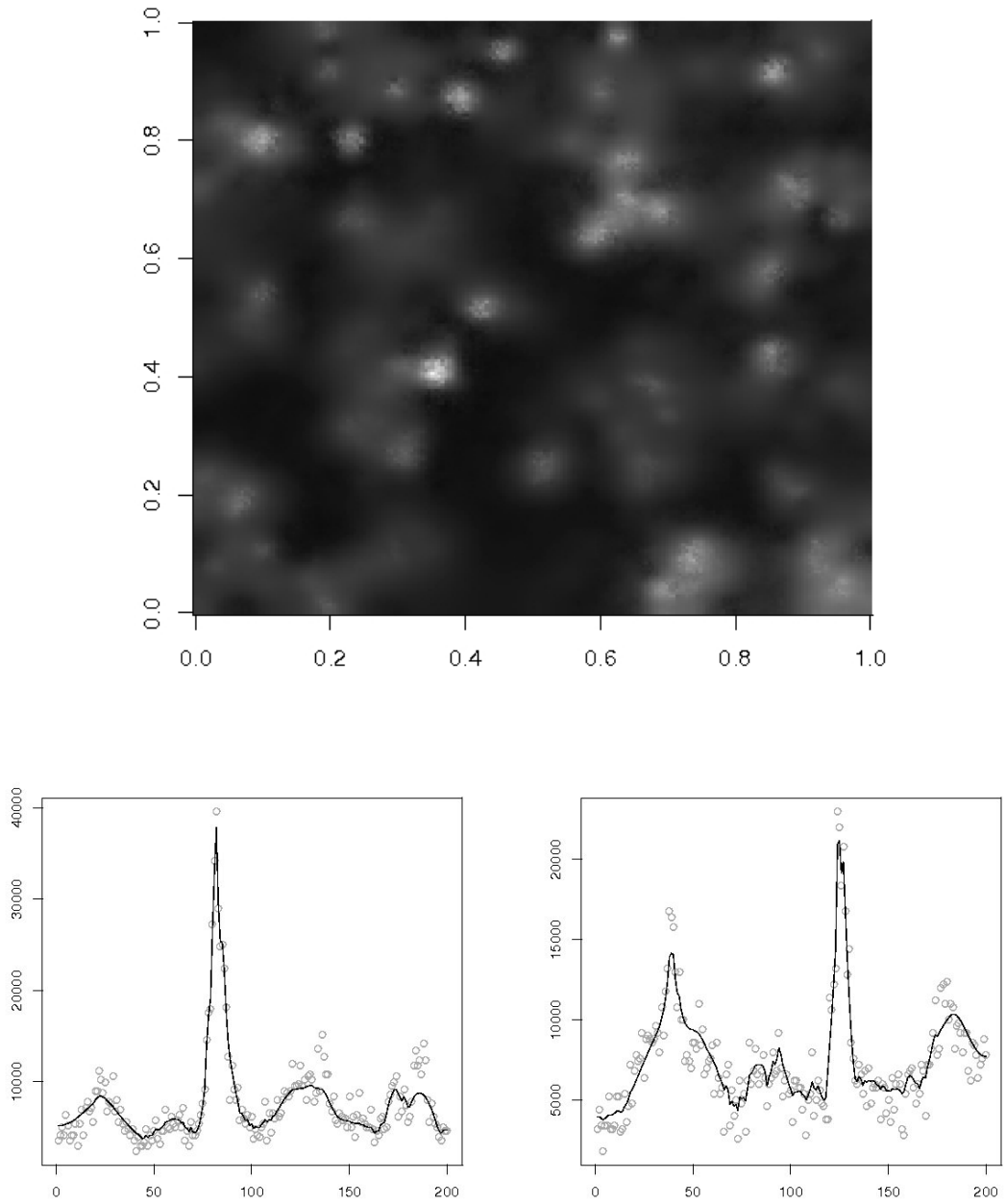


Figure 3.21: Diffusion estimator $\hat{f}_a$ for the Quantum Computing dataset with $\delta = 15$ and initial diffusivity $a = 200 \cdot \sigma_N$
Bottom: vertical and horizontal cuts

Chapter 4

Asymptotics of the Estimator $\hat{f}_\tau$

This chapter deals with the asymptotic behaviour of the diffusion estimator $\hat{f}_\tau$ with a global smoothing parameter τ. It will be shown that the estimator converges with the optimal rate both in one and two dimensions.

4.1 Asymptotics in the One-dimensional Case

Here we consider a function $f = f(x)$, $x \in [0,\, 1]$, which is three times differentiable, and $N = n + 1$ noisy observations at equidistant design points $x_i = \frac{i}{n}$:

$$Y_i = f(x_i) + Z_i \qquad \text{with } Z_i \sim \mathcal{N}(0,1) \text{ i.i.d. }, \quad i = 0, \ldots, n.$$

For the sake of simplicity the standard deviation σ of the noise is assumed to be 1.
The function f is to be estimated by $\hat{f}_\tau$, which is the solution of the homogeneous diffusion process stopped at the time τ. The starting values of the diffusion process are given by the data y_i. As seen in Chapter 2, the estimator $\hat{f}_\tau$ can be written in the integral form:

$$\hat{f}_\tau(x) = \frac{1}{\sqrt{4\pi\tau}} \int_{-\infty}^{\infty} Y(s) \exp\left(-\frac{(x-s)^2}{4\tau}\right) ds,$$

which yields with $Y(s) = f(s) + Z(s)$

$$\begin{aligned}\hat{f}_\tau(x) &= \frac{1}{\sqrt{4\pi\tau}} \int_{-\infty}^{\infty} (f(s) + Z(s)) \exp\left(-\frac{(x-s)^2}{4\tau}\right) ds \\ &= \frac{1}{\sqrt{4\pi\tau}} \int_{-\infty}^{\infty} f(s) \exp\left(-\frac{(x-s)^2}{4\tau}\right) ds + \frac{1}{\sqrt{4\pi\tau}} \int_{-\infty}^{\infty} Z(s) \exp\left(-\frac{(x-s)^2}{4\tau}\right) ds.\end{aligned}$$

Now the mean squared error of $\hat{f}_\tau$ is to be minimized. Therefore $f_\tau^\star$ and $Z_\tau^\star$ are defined in the following way:

$$f_\tau^\star(x) = \frac{1}{\sqrt{4\pi\tau}} \int_{-\infty}^{\infty} f(s) \exp\left(-\frac{(x-s)^2}{4\tau}\right) ds \quad \text{and}$$

$$Z_\tau^\star(x) = \frac{1}{\sqrt{4\pi\tau}} \int_{-\infty}^{\infty} Z(s) \exp\left(-\frac{(x-s)^2}{4\tau}\right) ds.$$

The quadratic error $|\hat{f}_\tau(x) - f(x)|^2$ is therefore given by

$$\left|\hat{f}_\tau(x) - f(x)\right|^2 = \left(f_\tau^\star(x) + Z_\tau^\star(x) - f(x)\right)^2 = \left((f_\tau^\star(x) - f(x)) + Z_\tau^\star(x)\right)^2$$

$$= \left(f_\tau^\star(x) - f(x)\right)^2 + 2(f_\tau^\star(x) - f(x))\, Z_\tau^\star(x) + \left(Z_\tau^\star(x)\right)^2,$$

and the mean squared error is

$$\mathbb{E}\left(\left(\hat{f}_\tau(x) - f(x)\right)^2\right) = \mathbb{E}\left(\left(f_\tau^\star(x) - f(x)\right)^2\right) + 2\,\mathbb{E}\left(\left(f_\tau^\star(x) - f(x)\right) \cdot Z_\tau^\star(x)\right) + \mathbb{E}\left(Z_\tau^\star(x)^2\right)$$

$$= \mathbb{E}\left(\left(f_\tau^\star(x) - f(x)\right)^2\right) + 2\,\mathbb{E}\left(f_\tau^\star(x) - f(x)\right) \cdot \mathbb{E}\left(Z_\tau^\star(x)\right) + \mathbb{E}\left(Z_\tau^\star(x)^2\right)$$

$$= \mathbb{E}\left(\left(f_\tau^\star(x) - f(x)\right)^2\right) + \mathbb{E}\left(Z_\tau^\star(x)^2\right),$$

as $f_\tau^\star(x) - f(x)$ und $Z_\tau^\star(x)$ are independent and $\mathbb{E}(Z_\tau^\star(x)) = 0$.

Now the bias $\mathbb{E}\left(\left(f_\tau^\star(x) - f(x)\right)^2\right)$ and the variance $\mathbb{E}\left(Z_\tau^\star(x)^2\right)$ are considered separately. $f(x)$ can be written as

$$f(x) = \frac{1}{\sqrt{4\pi\tau}} \int_{-\infty}^{\infty} f(x) \exp\left(-\frac{(x-s)^2}{4\tau}\right) ds$$

because of

$$\frac{1}{\sqrt{4\pi\tau}} \int_{-\infty}^{\infty} \exp\left(-\frac{(x-s)^2}{4\tau}\right) ds = \frac{1}{\sqrt{2\pi}\sqrt{2\tau}} \int_{-\infty}^{\infty} \exp\left(-\frac{(s-x)^2}{2(\sqrt{2\tau})^2}\right) ds$$

$$= \frac{1}{\sqrt{2\pi}\sigma} \int_{-\infty}^{\infty} \exp\left(-\frac{(s-\mu)^2}{2\sigma^2}\right) ds = \int_{-\infty}^{\infty} \Phi_{\mu,\sigma}(s)\, ds = 1 \quad \text{with } \mu = x \text{ and } \sigma = \sqrt{2\tau}.$$

So, by substituting $y = \frac{s-x}{\sqrt{4\tau}}$, one gets

$$f_\tau^\star(x) - f(x) = \frac{1}{\sqrt{4\pi\tau}} \int_{-\infty}^{\infty} (f(s) - f(x)) \exp\left(-\frac{(x-s)^2}{4\tau}\right) ds$$

$$= \frac{2\sqrt{\tau}}{\sqrt{4\pi\tau}} \int_{-\infty}^{\infty} (f(x + 2\sqrt{\tau}y) - f(x)) \exp(-y^2)\, dy$$

$$= \frac{1}{\sqrt{\pi}} \int_{-\infty}^{\infty} (f(x + 2\sqrt{\tau}y) - f(x)) \exp(-y^2)\, dy.$$

In order to estimate the difference $f(x + 2\sqrt{\tau}y) - f(x)$, the Taylor expansion of f around $x_0 = x$ is considered to obtain

$$f(x + 2\sqrt{\tau}y) = f(x) + f'(x)\, 2\sqrt{\tau}y + \frac{1}{2} f''(x)\, 4\tau y^2 + \frac{1}{6} f'''(\vartheta)\, 8\tau^{3/2} y^3$$

with $\vartheta \in [\, x,\, x + 2\sqrt{\tau}y \,]$. Substituting this expression in $f_\tau^\star(x) - f(x)$, one obtains

$$f_\tau^\star(x) - f(x) = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{\infty} \left(f'(x)\, 2\sqrt{\tau}y + \, 2\, f''(x)\, \tau y^2 + \frac{1}{6} f'''(\vartheta)\, 8\tau^{3/2} y^3 \right) \exp(-y^2)\, dy$$

$$= \frac{1}{\sqrt{\pi}} \int_{-\infty}^{\infty} \frac{1}{2} f''(x)\, 4\tau y^2 \, \exp(-y^2)\, dy = c' \cdot \tau$$

with a constant $c' = \frac{2}{\sqrt{\pi}} \int_{-\infty}^{\infty} f''(x)\, y^2 \, \exp(-y^2)\, dy$, since $\int_{-\infty}^{\infty} y \, \exp(-y^2)\, dy = \int_{-\infty}^{\infty} y^3 \, \exp(-y^2)\, dy = 0$. Therefore we have $\mathbb{E}\left((f_\tau^\star(x) - f(x))^2 \right) = c \cdot \tau^2$ for the bias with a constant $c = c'^2 > 0$.

Now the variance

$$Z_\tau^\star(x) = \frac{1}{\sqrt{4\pi\tau}} \int_{-\infty}^{\infty} Z(s) \exp\left(-\frac{(x-s)^2}{4\tau}\right) ds$$

is considered. Note that only the discrete values $Z_i = Z(x_i)$, where $x_i = \frac{i}{n}$, $i = 0, 1, \ldots, n$, are known. They satisfy the condition $Z_i \sim \mathcal{N}(0,1)$ i.i.d. The step function $\bar{Z}$ is defined by

$$\bar{Z}(s) = \sum_{i=0}^{n} Z_i \cdot \mathbb{1}_{\left(\frac{i}{n} - \frac{1}{2n},\, \frac{i}{n} + \frac{1}{2n}\right]}(s).$$

The

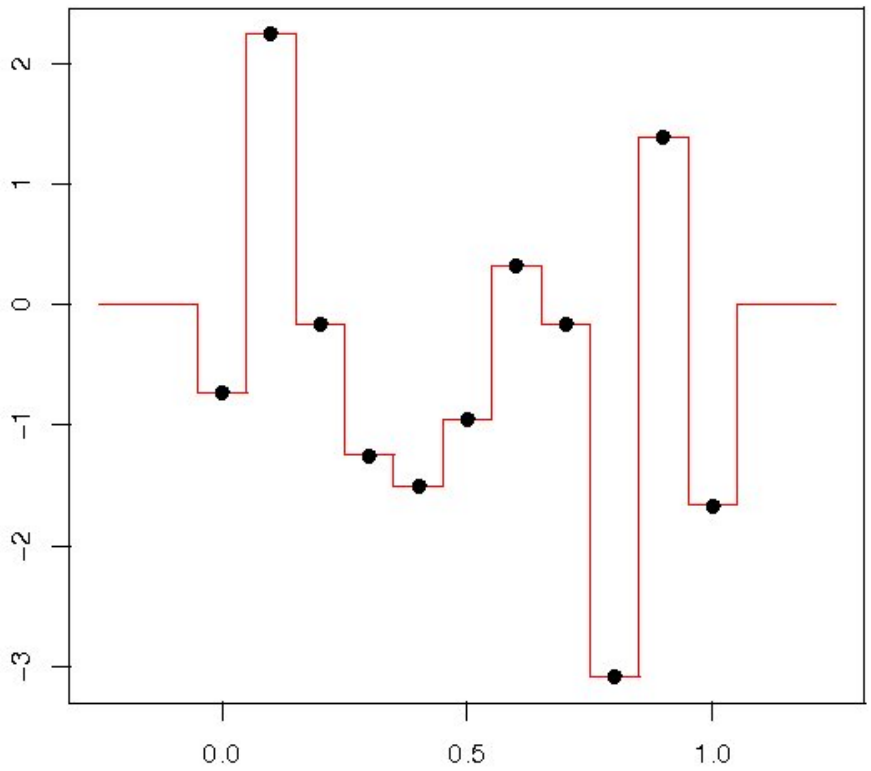


Figure 4.1: Step function $\bar{Z}$ (red line) generated by the values Z_i (black dots)

Substituting the step function $Z = \bar{Z}$ in $Z_\tau^\star$, one obtains

$$\mathbb{E}\left((Z_\tau^\star(x))^2\right) = \mathbb{E}\left(\frac{1}{4\pi\tau}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\bar{Z}(s)\,\bar{Z}(r)\exp\left(-\frac{(x-s)^2}{4\tau}\right)\exp\left(-\frac{(x-r)^2}{4\tau}\right)ds\,dr\right)$$

$$= \frac{1}{4\pi\tau}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\mathbb{E}\left(\bar{Z}(s)\,\bar{Z}(r)\exp\left(-\frac{(x-s)^2}{4\tau}\right)\exp\left(-\frac{(x-r)^2}{4\tau}\right)\right)ds\,dr$$

$$= \frac{1}{4\pi\tau}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\mathbb{E}\left(\bar{Z}(s)\,\bar{Z}(r)\right)\exp\left(-\frac{(x-s)^2}{4\tau}\right)\exp\left(-\frac{(x-r)^2}{4\tau}\right)ds\,dr.$$

Now we compute the expectation $\mathbb{E}\left(\bar{Z}(s)\,\bar{Z}(r)\right)$,

$$\mathbb{E}\left(\bar{Z}(s)\,\bar{Z}(r)\right) = \sum_{i=0}^{n}\sum_{j=0}^{n}\left(\mathbb{E}\left(Z_i\,Z_j\right)\mathbb{1}_{\left(\frac{i}{n}-\frac{1}{2n},\,\frac{i}{n}+\frac{1}{2n}\right]}(s)\;\mathbb{1}_{\left(\frac{j}{n}-\frac{1}{2n},\,\frac{j}{n}+\frac{1}{2n}\right]}(r)\right).$$

As $Z_0,\,Z_1,\ldots,\,Z_n \sim \mathcal{N}(0,1)$ are i.i.d., one has $\mathbb{E}(Z_i\,Z_j) = \mathbb{E}(Z_i^2) = \mathbb{V}(Z_i) = 1$ for $i = j$ and $\mathbb{E}(Z_i\,Z_j) = \mathbb{E}(Z_i)\,\mathbb{E}(Z_j) = 0$ for $i \neq j$. For the variance, this yields

$$\mathbb{E}\left(\left(Z_\tau^\star(x)\right)^2\right) = \frac{1}{4\pi\tau}\sum_{j=0}^{n}\int\limits_{\frac{j}{n}-\frac{1}{2n}}^{\frac{j}{n}+\frac{1}{2n}}\int\limits_{\frac{j}{n}-\frac{1}{2n}}^{\frac{j}{n}+\frac{1}{2n}}\exp\left(-\frac{(x-s)^2}{4\tau}\right)\exp\left(-\frac{(x-r)^2}{4\tau}\right)ds\,dr \quad (\star)$$

$$= \frac{1}{4\pi\tau}\sum_{j=0}^{n}\left(\int\limits_{\frac{j}{n}-\frac{1}{2n}}^{\frac{j}{n}+\frac{1}{2n}}\exp\left(-\frac{(x-s)^2}{4\tau}\right)ds\cdot\int\limits_{\frac{j}{n}-\frac{1}{2n}}^{\frac{j}{n}+\frac{1}{2n}}\exp\left(-\frac{(x-r)^2}{4\tau}\right)dr\right).$$

In order to estimate the variance, one considers

$$\int\limits_{\frac{j}{n}-\frac{1}{2n}}^{\frac{j}{n}+\frac{1}{2n}}\exp\left(-\frac{(x-s)^2}{4\tau}\right)ds \;=\; \int\limits_{-\frac{1}{2n}}^{\frac{1}{2n}}\exp\left(-\frac{(x-\frac{j}{n}-s)^2}{4\tau}\right)ds.$$

The Taylor expansion of the function $\exp(-\frac{x^2}{4\tau})$ around $x_0 = x - \frac{j}{n}$ is used with $s \in [-\frac{1}{2n}, \frac{1}{2n}]$ to yield, as $s \to 0$

$$\exp\left(-\frac{(x-\frac{j}{n}-s)^2}{4\tau}\right) = \exp\left(-\frac{(x-\frac{j}{n})^2}{4\tau}\right) + s\,\exp\left(-\frac{(x-\frac{j}{n}-s)^2}{4\tau}\right)\left(-\frac{x-\frac{j}{n}}{2\tau}\right) + o\left(s^2\right).$$

For the error this means, as $s \to 0$ and thus $n \to \infty$

$$\left|\exp\left(-\frac{(x-\frac{j}{n}-s)^2}{4\tau}\right) - \exp\left(-\frac{(x-\frac{j}{n})^2}{4\tau}\right)\right| \leq |s|\cdot\left|\frac{x-\frac{j}{n}}{2\tau}\cdot\exp\left(-\frac{(x-\frac{j}{n}-s)^2}{4\tau}\right)\right| + o(s^2)$$

$$\leq \frac{1}{2n}\cdot 1\cdot\frac{1+2j}{4n\tau} + o(n^{-2}) = c\cdot\frac{1}{n^2\tau} + o(n^{-2})$$
$$= \tau^{-1}\,O\left(n^{-2}\right) + o\left(n^{-2}\right) = \tau^{-1}\,O\left(n^{-2}\right).$$

So for the integral, one gets as $n \to \infty$:

$$\int\limits_{-\frac{1}{2n}}^{\frac{1}{2n}}\exp\left(-\frac{(x-\frac{j}{n}-s)^2}{4\tau}\right)ds \;=\; \int\limits_{-\frac{1}{2n}}^{\frac{1}{2n}}\exp\left(-\frac{(x-\frac{j}{n})^2}{4\tau}\right)ds + \frac{1}{n}\cdot\tau^{-1}\cdot O\left(n^{-2}\right)$$

$$= \int_{-\frac{1}{2n}}^{\frac{1}{2n}} \exp\left(-\frac{(x-\frac{j}{n})^2}{4\tau}\right) ds + \tau^{-1} O\left(n^{-3}\right) = \frac{1}{n} \cdot \exp\left(-\frac{(x-\frac{j}{n})^2}{4\tau}\right) + \tau^{-1} O\left(n^{-3}\right).$$

Inserting this expression into the double integral $(\star)$ yields:

$$\mathbb{E}\left((Z_\tau^\star(x))^2\right) = \frac{1}{4\pi\tau} \sum_{j=0}^{n} \left(\frac{1}{n} \exp\left(-\frac{(x-\frac{j}{n})^2}{4\tau}\right) \cdot \frac{1}{n} \exp\left(-\frac{(x-\frac{j}{n})^2}{4\tau}\right)\right) + \tau^{-1} O\left(n^{-3}\right)$$

$$= \frac{1}{4\pi\tau n} \frac{1}{n} \sum_{j=0}^{n} \exp\left(-\frac{(x-\frac{j}{n})^2}{2\tau}\right) + \tau^{-1} O\left(n^{-3}\right).$$

As $n \to \infty$, the Riemann sum converges to an integral, so one obtains

$$\mathbb{E}\left((Z_\tau^\star(x))^2\right) \approx \frac{1}{4\pi\tau n} \int_0^1 \exp\left(-\frac{(x-s)^2}{2\tau}\right) ds + \tau^{-1} O\left(n^{-3}\right)$$

$$\leq \frac{1}{4\pi\tau n} \int_{-\infty}^{\infty} \exp\left(-\frac{(x-s)^2}{2\tau}\right) ds + \tau^{-1} O\left(n^{-3}\right)$$

$$= \frac{1}{n\sqrt{\tau}} + \tau^{-1} O\left(n^{-3}\right) = \tau^{-\frac{1}{2}} O\left(n^{-1}\right) = \tau^{-\frac{1}{2}} O\left(N^{-1}\right).$$

One has the following orders of magnitude for the bias and the variance:

- $\mathbb{E}\left((f_\tau^\star(x) - f(x))^2\right) = c \cdot \tau^2 \quad \xrightarrow{\tau \to \infty} \quad \infty,$
- $\mathbb{E}\left((Z_\tau^\star(x))^2\right) = \tau^{-\frac{1}{2}} O\left(N^{-1}\right) \quad \xrightarrow{\tau \to \infty} \quad 0.$

In order to minimize the mean squared error, the orders of magnitude of τ^2 and $\tau^{-1/2} N^{-1}$ are set to be equal. This provides the asymptotically optimal value for the smoothing parameter $\tau = N^{-2/5}$, and altogether one has:

$$\mathbb{E}\left(\left(\hat{f}_\tau(x) - f(x)\right)^2\right) = O\left(N^{-4/5}\right).$$

Thus we have shown that the estimator $\hat{f}_\tau$ reaches the optimal rate of convergence $N^{-4/5}$ with $\tau = N^{-2/5}$.

4.2 Asymptotics in the Two-dimensional Case

The situation in the two-dimensional case is as follows: We assume that $N = (m+1)\cdot(n+1)$ noisy observations at $\mathbf{x}_{ij} = \left(\frac{i}{m}, \frac{j}{n}\right)$, $i = 0, \ldots, m$, $j = 0, \ldots, n$, are given for the function $f = f(\mathbf{x})$, $\mathbf{x} \in [0,1] \times [0,1]$, which is three times differentiable. Hence we have

$$Y_{ij} = f(\mathbf{x}_{ij}) + Z_{ij} \qquad \text{with } Z_{ij} \sim \mathcal{N}(0,1) \text{ i.i.d., } i = 0, \ldots, m,\ j = 0, \ldots, n.$$

In what follows, we only consider quadratic images for simplicity, i.e. we assume $m = n$.
The integral form of the estimator $\hat{f}_\tau$ in two dimensions is

$$\hat{f}_\tau(\mathbf{x}) = \frac{1}{4\pi\tau} \int\limits_{\mathbb{R}^2} Y(\mathbf{s}) \exp\left(-\frac{||\mathbf{x}-\mathbf{s}||^2}{4\tau}\right) d\mathbf{s}$$

$$= \frac{1}{4\pi\tau} \int\limits_{\mathbb{R}^2} (f(\mathbf{s}) + Z(\mathbf{s})) \exp\left(-\frac{||\mathbf{x}-\mathbf{s}||^2}{4\tau}\right) d\mathbf{s}.$$

In order to compute the mean squared error, the bias $\mathbb{E}\left((f_\tau^\star(\mathbf{x}) - f(\mathbf{x}))^2\right)$ and the variance $\mathbb{E}\left(Z_\tau^\star(\mathbf{x})^2\right)$ are considered separately. $f_\tau^\star$ and $Z_\tau^\star$ are defined in analogy to the one-dimensional case.
With $\frac{1}{4\pi\tau} \int\limits_{\mathbb{R}^2} \exp\left(-\frac{||\mathbf{x}-\mathbf{s}||^2}{4\tau}\right) d\mathbf{s} = 1$, the function f can be written as

$$f(\mathbf{x}) = \frac{1}{4\pi\tau} \int\limits_{\mathbb{R}^2} f(\mathbf{x}) \exp\left(-\frac{||\mathbf{x}-\mathbf{s}||^2}{4\tau}\right) d\mathbf{s},$$

and thus

$$f_\tau^\star(\mathbf{x}) - f(\mathbf{x}) = \frac{1}{4\pi\tau} \int\limits_{\mathbb{R}^2} (f(\mathbf{s}) - f(\mathbf{x})) \exp\left(-\frac{||\mathbf{x}-\mathbf{s}||^2}{4\tau}\right) d\mathbf{s}.$$

Here we substitute y for $\frac{\mathbf{s}-\mathbf{x}}{\sqrt{4\tau}}$. Since the Jacobian matrix of the mapping $\mathbf{s} \mapsto \frac{\mathbf{s}-\mathbf{x}}{\sqrt{4\tau}}$ is

$$\begin{pmatrix} \frac{1}{\sqrt{4\tau}} & 0 \\ 0 & \frac{1}{\sqrt{4\tau}} \end{pmatrix}$$

with determinant $\frac{1}{4\tau}$, one obtains

$$f_\tau^\star(\mathbf{x}) - f(\mathbf{x}) = \frac{1}{\pi} \int\limits_{\mathbb{R}^2} (f(\mathbf{x} + 2\sqrt{\tau}\mathbf{y}) - f(\mathbf{x})) \exp(-||\mathbf{y}||^2)\, d\mathbf{y}. \qquad (\star\star)$$

We again use the Taylor expansion of f around $\mathbf{x}_0 = \mathbf{x}$ to get

$$f(\mathbf{x} + 2\sqrt{\tau}\mathbf{y})$$

$$= f(\mathbf{x}) + \sum_{|\alpha|=1} \frac{D^\alpha f(\mathbf{x})}{\alpha!}(2\sqrt{\tau}\mathbf{y})^\alpha + \sum_{|\alpha|=2} \frac{D^\alpha f(\mathbf{x})}{\alpha!}(2\sqrt{\tau}\mathbf{y})^\alpha + \sum_{|\alpha|=3} \frac{D^\alpha f(\mathbf{x}+\theta 2\sqrt{\tau}\mathbf{y})}{\alpha!}(2\sqrt{\tau}\mathbf{y})^\alpha$$

with $\theta \in [0,1]$. The sums in the above equation are

$$\sum_{|\alpha|=1} \frac{D^\alpha f(\mathbf{x})}{\alpha!}(2\sqrt{\tau}\mathbf{y})^\alpha = \frac{\partial f}{\partial x_1}(\mathbf{x})\,(2\sqrt{\tau}y_1) + \frac{\partial f}{\partial x_2}(\mathbf{x})\,(2\sqrt{\tau}y_2)\,,$$

$$\sum_{|\alpha|=2} \frac{D^\alpha f(\mathbf{x})}{\alpha!}(2\sqrt{\tau}\mathbf{y})^\alpha = \frac{1}{2}\,\frac{\partial^2 f}{\partial x_1^2}(\mathbf{x})\,(2\sqrt{\tau}y_1)^2$$

$$+\frac{1}{2}\,\frac{\partial^2 f}{\partial x_2^2}(\mathbf{x})\,(2\sqrt{\tau}y_2)^2 + \frac{\partial^2 f}{\partial x_1\,x_2}(\mathbf{x})\,(2\sqrt{\tau}y_1)(2\sqrt{\tau}y_2) \quad \text{and}$$

$$\sum_{|\alpha|=3} \frac{D^\alpha f(\mathbf{x}+\theta 2\sqrt{\tau}\mathbf{y})}{\alpha!}(2\sqrt{\tau}\mathbf{y})^\alpha$$

$$= \frac{1}{6}\,\frac{\partial^3 f}{\partial x_1^3}(\mathbf{x}+\theta 2\sqrt{\tau}\mathbf{y})\,(2\sqrt{\tau}y_1)^3 + \frac{1}{6}\,\frac{\partial^3 f}{\partial x_2^3}(\mathbf{x}+\theta 2\sqrt{\tau}\mathbf{y})\,(2\sqrt{\tau}y_2)^3$$

$$+\frac{1}{2}\,\frac{\partial^3 f}{\partial x_1^2\,x_2}(\mathbf{x}+\theta 2\sqrt{\tau}\mathbf{y})\,(2\sqrt{\tau}y_1)^2(2\sqrt{\tau}y_2)$$

$$+\frac{1}{2}\,\frac{\partial^3 f}{\partial x_1\,x_2^2}(\mathbf{x}+\theta 2\sqrt{\tau}\mathbf{y})(2\sqrt{\tau}y_1)\,(2\sqrt{\tau}y_2)^2.$$

In the expression $(\star\star)$, the difference $f(\mathbf{x}+2\sqrt{\tau}\mathbf{y}) - f(\mathbf{x})$ is multiplied by $\exp(-||\mathbf{y}||^2)$ and integrated over $\mathbb{R}^2$. Therefore the following integrals have to be considered for $|\alpha| \le 3$:

$$\int\limits_{\mathbb{R}^2} \frac{D^\alpha f(\mathbf{x})}{\alpha!}(2\sqrt{\tau}\mathbf{y})^\alpha\, \exp(-||\mathbf{y}||^2) d\mathbf{y}$$

$$= \int\limits_{-\infty}^{\infty}\int\limits_{-\infty}^{\infty} \frac{1}{\alpha_1!\,\alpha_2!}\,\frac{\partial^{\alpha_1+\alpha_2} f}{\partial x_1^{\alpha_1}\partial x_2^{\alpha_2}}(\mathbf{x})\,(2\sqrt{\tau}y_1)^{\alpha_1}\exp(-y_1^2)\;(2\sqrt{\tau}y_2)^{\alpha_2}\exp(-y_2^2)\,dy_1\,dy_2$$

$$= \frac{1}{\alpha_1!\,\alpha_2!}\,\frac{\partial^{\alpha_1+\alpha_2} f}{\partial x_1^{\alpha_1}\partial x_2^{\alpha_2}}\,(\mathbf{x}) \int\limits_{-\infty}^{\infty} (2\sqrt{\tau}y_1)^{\alpha_1}\exp(-y_1^2)\,dy_1 \int\limits_{-\infty}^{\infty} (2\sqrt{\tau}y_2)^{\alpha_2}\exp(-y_2^2)\,dy_2.$$

They are equal to 0, if at least one of the two numbers α_1 or α_2 is odd. This is accomplished for $|\alpha| = 1$, $|\alpha| = 3$ and $\alpha = (1,1)$, and one gets

$$f_\tau^\star(\mathbf{x}) - f(\mathbf{x}) = \frac{1}{\pi} \int\limits_{\mathbb{R}^2} \frac{1}{2} \left(\frac{\partial^2 f}{\partial x_1^2}(\mathbf{x}) \, (2\sqrt{\tau} y_1)^2 + \frac{\partial^2 f}{\partial x_2^2}(\mathbf{x}) \, (2\sqrt{\tau} y_2)^2 \right) \exp(-||\mathbf{y}||^2) \, d\mathbf{y}$$

$$= \frac{2\tau}{\pi} \int\limits_{\mathbb{R}^2} \left(\frac{\partial^2 f}{\partial x_1^2}(\mathbf{x}) \, y_1^2 + \frac{\partial^2 f}{\partial x_2^2}(\mathbf{x}) \, y_2^2 \right) \exp(-||\mathbf{y}||^2) \, d\mathbf{y} = c' \cdot \tau$$

with a constant $c' = \frac{2}{\pi} \int\limits_{\mathbb{R}^2} \left(\frac{\partial^2 f}{\partial x_1^2}(\mathbf{x}) \, y_1^2 + \frac{\partial^2 f}{\partial x_2^2}(\mathbf{x}) \, y_2^2 \right) \exp(-||\mathbf{y}||^2) \, d\mathbf{y}$. Hence the bias is given as $\mathbb{E}\left((f_\tau^\star(\mathbf{x}) - f(\mathbf{x}))^2 \right) = c \cdot \tau^2$ with a constant $c = c'^2 > 0$.

Now the variance

$$Z_\tau^\star(\mathbf{x}) = \frac{1}{4\pi\tau} \int\limits_{\mathbb{R}^2} Z(\mathbf{s}) \exp\left(-\frac{||\mathbf{x} - \mathbf{s}||^2}{4\tau} \right) d\mathbf{s}$$

has to be considered. Only the discrete values Z_{ij} at $\mathbf{x}_{ij} = (\frac{i}{n}, \frac{j}{n})$, $i, j = 0, \ldots, n$, with $Z_{ij} \sim \mathcal{N}(0,1)$ i.i.d. are given for the function Z, and the two-dimensional step function $\bar{Z}$ is generated by these values. Note that $\bar{Z}$ is constant and has the value Z_{ij} on each square $Q_{ij} = \left(\frac{i}{n} - \frac{1}{2n}, \frac{i}{n} + \frac{1}{2n}\right] \times \left(\frac{j}{n} - \frac{1}{2n}, \frac{j}{n} + \frac{1}{2n}\right]$, i.e.

$$\bar{Z}(\mathbf{s}) = \sum_{i=0}^{n} \sum_{j=0}^{n} Z_{ij} \cdot \mathbb{1}_{\left(\frac{i}{n} - \frac{1}{2n}, \frac{i}{n} + \frac{1}{2n}\right] \times \left(\frac{j}{n} - \frac{1}{2n}, \frac{j}{n} + \frac{1}{2n}\right]}(\mathbf{s}).$$

Substituting the step function $\bar{Z}$ into $Z_\tau^\star$, one obtains

$$\mathbb{E}\left((Z_\tau^\star(\mathbf{x}))^2 \right)$$

$$= \mathbb{E}\left(\frac{1}{16\pi^2\tau^2} \int\limits_{\mathbb{R}^2} \int\limits_{\mathbb{R}^2} \bar{Z}(\mathbf{s}) \, \bar{Z}(\mathbf{r}) \exp\left(-\frac{||\mathbf{x} - \mathbf{s}||^2}{4\tau} \right) \exp\left(-\frac{||\mathbf{x} - \mathbf{r}||^2}{4\tau} \right) d\mathbf{s} \, d\mathbf{r} \right)$$

$$= \frac{1}{16\pi^2\tau^2} \int\limits_{\mathbb{R}^2} \int\limits_{\mathbb{R}^2} \mathbb{E}\left(\bar{Z}(\mathbf{s}) \, \bar{Z}(\mathbf{r}) \right) \exp\left(-\frac{||\mathbf{x} - \mathbf{s}||^2}{4\tau} \right) \exp\left(-\frac{||\mathbf{x} - \mathbf{r}||^2}{4\tau} \right) d\mathbf{s} \, d\mathbf{r}.$$

Now we compute $\mathbb{E}\left(\bar{Z}(\mathbf{s})\,\bar{Z}(\mathbf{r})\right)$ by putting

$$\mathbb{E}\left(\bar{Z}(\mathbf{s})\,\bar{Z}(\mathbf{r})\right)$$

$$= \sum_{i,j,k,l=0}^{n} \left(\mathbb{E}\left(Z_{i,j}\,Z_{k,l}\right) \mathbb{1}_{\left(\frac{i}{n}-\frac{1}{2n},\,\frac{i}{n}+\frac{1}{2n}\right]\times\left(\frac{j}{n}-\frac{1}{2n},\,\frac{j}{n}+\frac{1}{2n}\right]}(\mathbf{s})\;\mathbb{1}_{\left(\frac{k}{n}-\frac{1}{2n},\,\frac{k}{n}+\frac{1}{2n}\right]\times\left(\frac{l}{n}-\frac{1}{2n},\,\frac{l}{n}+\frac{1}{2n}\right]}(\mathbf{r})\right).$$

As the random variables $Z_{ij} \sim \mathcal{N}(0,1)$ are i.i.d., one has $\mathbb{E}(Z_{ij}\,Z_{kl}) = \mathbb{E}(Z_{ij}^2) = \mathbb{V}(Z_{ij}) = 1$ for $(i,j) = (k,l)$ and $\mathbb{E}(Z_{ij}\,Z_{kl}) = \mathbb{E}(Z_{ij})\,\mathbb{E}(Z_{kl}) = 0$ for $(i,j) \neq (k,l)$. Hence for the variance one gets

$$\mathbb{E}\left((Z_\tau^\star(x))^2\right) = \frac{1}{16\pi^2\tau^2} \sum_{i,j=0}^{n} \int\limits_{Q_{i,j}} \int\limits_{Q_{i,j}} \exp\left(-\frac{||\mathbf{x}-\mathbf{s}||^2}{4\tau}\right) \exp\left(-\frac{||\mathbf{x}-\mathbf{r}||^2}{4\tau}\right) d\mathbf{s}\,d\mathbf{r} \quad (\star\star\star)$$

with

$$\int\limits_{Q_{i,j}} \int\limits_{Q_{i,j}} \exp\left(-\frac{||\mathbf{x}-\mathbf{s}||^2}{4\tau}\right) \exp\left(-\frac{||\mathbf{x}-\mathbf{r}||^2}{4\tau}\right) d\mathbf{s}\,d\mathbf{r}$$

$$= \int\limits_{\frac{i}{n}-\frac{1}{2n}}^{\frac{i}{n}+\frac{1}{2n}} \int\limits_{\frac{j}{n}-\frac{1}{2n}}^{\frac{j}{n}+\frac{1}{2n}} \int\limits_{\frac{i}{n}-\frac{1}{2n}}^{\frac{i}{n}+\frac{1}{2n}} \int\limits_{\frac{j}{n}-\frac{1}{2n}}^{\frac{j}{n}+\frac{1}{2n}} \exp\left(-\frac{||\mathbf{x}-\mathbf{s}||^2}{4\tau}\right) \exp\left(-\frac{||\mathbf{x}-\mathbf{r}||^2}{4\tau}\right) ds_2\,ds_1\,dr_2\,dr_1$$

$$= \int\limits_{\frac{i}{n}-\frac{1}{2n}}^{\frac{i}{n}+\frac{1}{2n}} \exp\left(-\frac{(x_1-s_1)^2}{4\tau}\right) ds_1 \cdot \int\limits_{\frac{j}{n}-\frac{1}{2n}}^{\frac{j}{n}+\frac{1}{2n}} \exp\left(-\frac{(x_2-s_2)^2}{4\tau}\right) ds_2 \cdot$$

$$\int\limits_{\frac{i}{n}-\frac{1}{2n}}^{\frac{i}{n}+\frac{1}{2n}} \exp\left(-\frac{(x_1-r_1)^2}{4\tau}\right) dr_1 \cdot \int\limits_{\frac{j}{n}-\frac{1}{2n}}^{\frac{j}{n}+\frac{1}{2n}} \exp\left(-\frac{(x_2-r_2)^2}{4\tau}\right) dr_2\,.$$

At this point we use the estimate

$$\int\limits_{\frac{i}{n}-\frac{1}{2n}}^{\frac{i}{n}+\frac{1}{2n}} \exp\left(-\frac{(z-t)^2}{4\tau}\right) dt = \int\limits_{-\frac{1}{2n}}^{\frac{1}{2n}} \exp\left(-\frac{(z-\frac{i}{n}-t)^2}{4\tau}\right) dt$$

$$= \frac{1}{n} \cdot \exp\left(-\frac{(z-\frac{i}{n})^2}{4\tau}\right) + \tau^{-1}\,O\left(n^{-3}\right) \quad \text{as } n \to \infty,$$

which has been derived in the one-dimensional case. Applying it to the double integral $(\star\star\star)$, one obtains as $n \to \infty$:

$$\mathbb{E}\left(\left(Z_\tau^\star(\mathbf{x})\right)^2\right)$$

$$= \frac{1}{16\pi^2\tau^2}\left(\frac{1}{n^4}\sum_{i,j=0}^{n}\left(\exp\left(-\frac{(x_1-\frac{i}{n})^2}{2\tau}\right)\exp\left(-\frac{(x_2-\frac{j}{n})^2}{2\tau}\right)\right) + \tau^{-1}\,O\left(n^{-3}\right)\right)$$

$$= \frac{1}{16\pi^2\tau^2n^2}\,\frac{1}{n}\sum_{i=0}^{n}\exp\left(-\frac{(x_1-\frac{i}{n})^2}{2\tau}\right)\frac{1}{n}\sum_{j=0}^{n}\exp\left(-\frac{(x_2-\frac{j}{n})^2}{2\tau}\right) + \tau^{-3}\,O\left(n^{-3}\right)$$

$$\approx \frac{1}{16\pi^2\tau^2n^2}\int_0^1\exp\left(-\frac{(x_1-s_1)^2}{2\tau}\right)ds_1\int_0^1\exp\left(-\frac{(x_2-s_2)^2}{2\tau}\right)ds_2 + \tau^{-3}\,O\left(n^{-3}\right)$$

$$\leq \frac{1}{16\pi^2\tau^2n^2}\int_{-\infty}^{\infty}\exp\left(-\frac{(x_1-s_1)^2}{2\tau}\right)ds_1\int_{-\infty}^{\infty}\exp\left(-\frac{(x_2-s_2)^2}{2\tau}\right)ds_2 + \tau^{-3}\,O\left(n^{-3}\right)$$

$$= \frac{1}{8\pi^2\tau n^2} + \tau^{-3}\,O\left(n^{-3}\right) = \tau^{-1}\,O\left(n^{-2}\right) = \tau^{-1}\,O\left(N^{-1}\right).$$

Thus the bias and the variance are of the following orders of magnitude:

- $\mathbb{E}\left(\left(f_\tau^\star(x)-f(x)\right)^2\right) = c\cdot\tau^2 \quad \xrightarrow{\tau\to\infty} \quad \infty,$
- $\mathbb{E}\left(\left(Z_\tau^\star(x)\right)^2\right) = \tau^{-1}\,O\left(N^{-1}\right) \quad \xrightarrow{\tau\to\infty} \quad 0.$

In order to get the minimal mean squared error, the orders of magnitude of τ^2 and $\tau^{-1}N^{-1}$ are set to be equal as in the one-dimensional case. One obtains $\tau = N^{-1/3}$ as the asymptotically optimal value for the smoothing parameter. Altogether one has:

$$\mathbb{E}\left(\left(\hat{f}_\tau(x)-f(x)\right)^2\right) = O\left(N^{-2/3}\right).$$

Hence the estimator $\hat{f}_\tau$ converges for $\tau = N^{-1/3}$ with the optimal rate $N^{-2/3}$.

Chapter 5

Theoretical Background for the Multiresolution Criterion

In this chapter, we present some theoretical results for the Brownian motion and its two-dimensional analogue, the Brownian sheet. These results concerning the modulus of continuity of the Brownian motion and the Brownian sheet justify the multiresolution criterion via the embedding of Gaussian white noise into the Brownian motion in one dimension, and the Brownian sheet in two dimensions.

For the proofs in this chapter, we need the following lemmas, which we state without proof. For the proof of Lemma 5.1 we refer to Pollard [Pol84], Appendix B. The proof of Lemma 5.2 is straightforward.

Lemma 5.1 *Let X be a normally distributed random variable, $X \sim \mathcal{N}(0,1)$. Then the following inequalities hold:*

$$\mathbb{P}\left(|X| \geq \lambda\right) \leq 2 \exp\left(-\frac{1}{2}\lambda^2\right) \quad \textit{and}$$

$$\mathbb{P}\left(|X| \geq \lambda\right) \leq \sqrt{\frac{2}{\pi}}\,\frac{1}{\lambda} \exp\left(-\frac{1}{2}\lambda^2\right).$$

Lemma 5.2

1. *For any $\delta \geq 0$, the function $\varphi_\delta : (0,1] \to \mathbb{R}$, which is defined by*

$$\varphi_\delta(q) = \sqrt{q}\sqrt{\delta \log \frac{1}{q}},$$

is monotonously increasing for $q \in (0, \alpha_0)$ with some $\alpha_0 \in (0,1)$.

2. *For any $\gamma \geq 0$, the function $\psi_\gamma : (0,1] \to \mathbb{R}$, which is defined by*

$$\psi_\gamma(q) = \sqrt{q}\sqrt{2\log\frac{1}{q} + \gamma \log\log\frac{1}{q}},$$

is monotonously increasing for $q \in (0, \beta_0)$ with some $\beta_0 \in (0,1)$.

5.1 The Modulus of Continuity of the Brownian Motion

The modulus of continuity of the Brownian motion is first being considered by Paul Lévy. The following theorem is an extension of Lévy's results.

Theorem 5.3 *Let $B = B(s)$ be a standard Brownian motion.*

1. *For all $\varepsilon > 0$, there exists $h_0 = h_0(\varepsilon) \in (0,1)$, so that for all $\delta > 2$*

$$\mathbb{P}\left(\sup_{0\leq s<t\leq 1;\, t-s\leq h_0} \frac{|B(t)-B(s)|}{\sqrt{t-s}\sqrt{\delta \log\frac{1}{t-s}}} \leq 1\right) \geq 1-\varepsilon.$$

2. *For all $\varepsilon > 0$, there exists $h_0 = h_0(\varepsilon) \in (0,1)$, so that for all $\gamma > 5$*

$$\mathbb{P}\left(\sup_{0\leq s<t\leq 1;\, t-s\leq h_0} \frac{|B(t)-B(s)|}{\sqrt{t-s}\sqrt{2\log\frac{1}{t-s} + \gamma\log\log\frac{1}{t-s}}} \leq 1\right) \geq 1-\varepsilon.$$

3. *For all $\varepsilon > 0$, there exists $\delta_0 = \delta_0(\varepsilon) > 2$, so that for all $\delta > \delta_0$*

$$\mathbb{P}\left(\sup_{0\leq s<t\leq 1;\, t-s\neq 1} \frac{|B(t)-B(s)|}{\sqrt{t-s}\sqrt{\delta \log\frac{1}{t-s}}} \leq 1\right) \geq 1-\varepsilon.$$

4. *For all $\varepsilon > 0$, there exists $\gamma_0 = \gamma_0(\varepsilon) > 0$, so that for all $\gamma > \gamma_0$*

$$\mathbb{P}\left(\sup_{0\leq s<t\leq 1;\, t-s\neq 1} \frac{|B(t)-B(s)|}{\sqrt{t-s}\sqrt{2\log\frac{1}{t-s} + \gamma\log\log\frac{1}{t-s}}} \leq 1\right) \geq 1-\varepsilon.$$

Proof
1., 2. For the proof of parts 1 and 2 we refer to Lévy [Lév54] and Sirao [Sir54], respectively.

3. In order to prove part 3, we use part 1 with $\frac{\varepsilon}{2}$, hence we have for $h_0 = h_0(\frac{\varepsilon}{2})$ and for all $\delta > 2$

$$\mathbb{P}\left(\sup_{0\le s<t\le 1;\, t-s\le h_0} \frac{|B(t)-B(s)|}{\sqrt{t-s}\sqrt{\delta \log \frac{1}{t-s}}} > 1 \right) \le \frac{\varepsilon}{2}.$$

Therefore it remains to show that there exists $\delta_0 = \delta_0(\varepsilon) > 2$, such that for all $\delta > \delta_0$

$$\mathbb{P}\left(\sup_{0\le s<t\le 1;\, h_0<t-s<1} \frac{|B(t)-B(s)|}{\sqrt{t-s}\sqrt{\delta \log \frac{1}{t-s}}} > 1 \right) \le \frac{\varepsilon}{2}.$$

In the proof of part 1 the interval $I = [s,t]$ is approximated by an interval I_m of the form $I_m = [k\,2^{-m}, (k+l)\,2^{-m}]$ with $m \in \mathbb{N}$, $1 \le k < 2^m$, $1 \le l \le m$. Here the condition $h \le h_0$ corresponds to the condition $m \ge m_0$. Thus considering the intervals $I = [s,t]$ with $t - s > h_0$, we have $m < m_0$ for the corresponding approximation I_m. Let

$$\mathcal{I}_{m_0}^- := \{I_m = [k\,2^{-m}, (k+l)\,2^{-m}],\ 1 \le m < m_0,\ 1 \le k < 2^m,\ 1 \le l \le m\}$$

be the set of all approximating intervals for the intervals $I = [s,t]$ with $h_0 < t - s < 1$. Then $\#\mathcal{I}_{m_0}^- =: A = A(h_0) = A(\varepsilon)$ is a finite number. As

$$\mathbb{P}\left(\sup_{0\le s<t\le 1:\, h_0<t-s<1} \frac{|B(t)-B(s)|}{\sqrt{t-s}\sqrt{\delta \log \frac{1}{t-s}}} > 1 \right)$$

is reduced to

$$\mathbb{P}\left(\sup_{[s_m,t_m]\in\mathcal{I}_{m_0}^-} \frac{|B(t_m)-B(s_m)|}{\sqrt{t_m-s_m}\sqrt{\delta \log \frac{1}{t_m-s_m}}} > 1 \right),$$

we can use the inequalities

$$\mathbb{P}\left(\sup_{[s_m,t_m]\in\mathcal{I}_{m_0}^-} \frac{|B(t_m)-B(s_m)|}{\sqrt{t_m-s_m}\sqrt{\delta \log \frac{1}{t_m-s_m}}} > 1 \right)$$

$$\le \#\mathcal{I}_{m_0}^- \cdot \mathbb{P}\left(\frac{|B(t_m)-B(s_m)|}{\sqrt{t_m-s_m}} > \sqrt{\delta \log \frac{1}{t_m-s_m}} \right)$$

$$\leq A(\varepsilon) \cdot \mathbb{P}\left(\frac{|B(t_m) - B(s_m)|}{\sqrt{t_m - s_m}} > \sqrt{\delta \log 2} \right) \qquad \text{as } t_m - s_m \leq 2^{-1}.$$

For $\frac{B(t_m)-B(s_m)}{\sqrt{t_m-s_m}} \sim \mathcal{N}(0,1)$, we can use Lemma 5.1 to obtain

$$A(\varepsilon) \cdot \mathbb{P}\left(\frac{|B(t_m) - B(s_m)|}{\sqrt{t_m - s_m}} > \sqrt{\delta \log 2} \right) \leq A(\varepsilon) \cdot \exp\left(-\frac{\delta}{2} \log 2 \right).$$

Hence if

$$A(\varepsilon) \cdot \exp\left(-\frac{\delta}{2} \log 2 \right) \leq \frac{\varepsilon}{2}, \quad \text{i.e. if } \delta \geq -\frac{2}{\log 2} \log \frac{\varepsilon}{2A(\varepsilon)},$$

we obtain

$$\mathbb{P}\left(\sup_{0 \leq s < t \leq 1 \colon h_0 < t-s < 1} \frac{|B(t) - B(s)|}{\sqrt{t-s}\sqrt{\delta \log \frac{1}{t-s}}} > 1 \right) \leq \frac{\varepsilon}{2},$$

which proves part 3.

4. Part 4 can be shown to hold by the arguments used in the proof of part 3.

□

We note that the condition $t - s \neq 1$ in parts 3 and 4 is only necessary to ensure $\log \frac{1}{t-s} \neq 0$. If one wants to include the interval $[s,t] = [0,1]$, this can be achieved by adding a constant $c_1 > 1$ to have $\log \frac{c_1}{t-s} \neq 0$, and a constant $c_2 > e$ to have $\log \log \frac{c_2}{t-s} \neq 0$. For instance with $e > 1$ and $e^e > e$ we obtain

3′. For all $\varepsilon > 0$, there exists $\delta_0 = \delta_0(\varepsilon) > 2$, so that for all $\delta > \delta_0$

$$\mathbb{P}\left(\sup_{0 \leq s < t \leq 1} \frac{|B(t) - B(s)|}{\sqrt{t-s}\sqrt{\delta \log \frac{e}{t-s}}} \leq 1 \right) \geq 1 - \varepsilon.$$

4′. For all $\varepsilon > 0$, there exists $\gamma_0 = \gamma_0(\varepsilon) > 0$, so that for all $\gamma > \gamma_0$

$$\mathbb{P}\left(\sup_{0 \leq s < t \leq 1} \frac{|B(t) - B(s)|}{\sqrt{t-s}\sqrt{2 \log \frac{1}{t-s} + \gamma \log \log \frac{e^e}{t-s}}} \leq 1 \right) \geq 1 - \varepsilon.$$

5.2 The Modulus of Continuity of the Brownian Sheet

This section deals with the two-dimensional analogue of the Brownian motion, namely the Brownian sheet, which is defined as follows:

Definition 5.4

1. *A stochastic process B on the unit square is called a two-dimensional Brownian sheet, if for $s = (s_1, s_2) \in [0,1]^2$ and $t = (t_1, t_2) \in [0,1]^2$*
   - $\mathbb{E}(B(s)) = 0$,
   - $Cov(B(s),\, B(t)) = \min(s_1, t_1) \cdot \min(s_2, t_2)$.
2. *Let B be a Brownian sheet and $R = [s_1, t_1] \times [s_2, t_2]$ be a rectangle within the unit square. Then we define*

$$B(R) := B(t_1, t_2) - B(t_1, s_2) - B(s_1, t_2) + B(s_1, s_2).$$

Let $|R|$ be the area of R. Then the above defined random variables $B(R)$ are normally distributed with variance $|R|$, $B(R) \sim \mathcal{N}(0, |R|)$, and they are independent for disjoint rectangles R_1 and R_2.

The next definition contains some useful notations for rectangles and for wedges, which are defined in Section 3.4.2, as well as some constructions needed for the formulation of the Theorem 5.9 about wedges.

Definition 5.5

1. *The set of all rectangles in the unit square is denoted by $\mathcal{R}$.*
2. *A rectangle $R = [s_1, s_1 + h_1] \times [s_2, s_2 + h_2] \in \mathcal{R}$ is called a κ-regular rectangle for $0 < \kappa \leq 1$, if $-\log h_1 \geq (-\log h_2)^\kappa \geq (-\log h_1)^\kappa$ or $-\log h_2 \geq (-\log h_1)^\kappa \geq (-\log h_2)^\kappa$.*
   The set of all κ-regular rectangles in the unit square is denoted by $\mathcal{R}^{\kappa-reg}$.
3. *The set of all wedges in the unit square is denoted by $\mathcal{W}$.*
4. *For any wedge $W \in \mathcal{W}$, the corresponding rectangle, denoted by $R(W)$, is defined as the smallest $R \in \mathcal{R}$ with $W \subseteq R$. The side lengths of $R(W)$ are denoted by h_1 and h_2.*
5. *A wedge $W \in \mathcal{W}$ is called a κ-regular wedge for $0 < \kappa \leq 1$, if the corresponding rectangle $R(W)$ is κ-regular. The set of all κ-regular wedges is denoted by $\mathcal{W}^{\kappa-reg}$.*

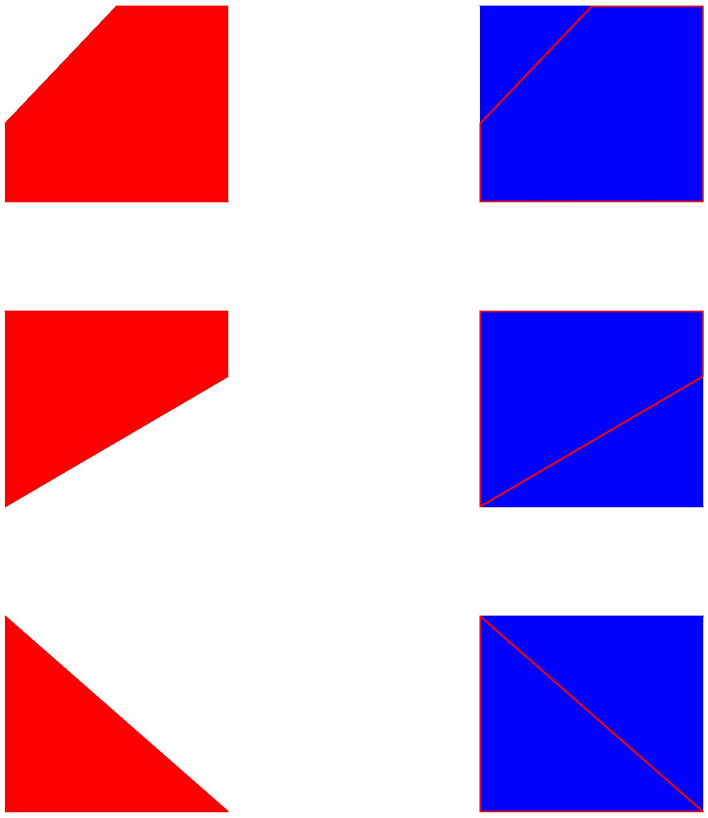


Figure 5.1: Three types of wedges W and corresponding rectangles $R(W)$

Now we can proceed to prove the main result of this section concerning the modulus of continuity for the Brownian sheet. At first, it is formulated for rectangles and secondly for wedges.

5.2.1 The Modulus of Continuity for Rectangles

Theorem 5.6 *Let $B = B(s_1, s_2)$ be a Brownian sheet.*

1. *For all $\varepsilon > 0$, there exists $q_0 = q_0(\varepsilon) \in (0, \alpha_0)$, so that for all $\delta > 2$*

$$\mathbb{P}\left(\sup_{R\in\mathcal{R};\, h_1,h_2\le\frac{1}{2},\, |R|\le q_0} \frac{|B(R)|}{\sqrt{|R|}\sqrt{\delta \log \frac{1}{|R|}}} \le 1 \right) \ge 1 - \varepsilon.$$

2. *For all $\varepsilon > 0$ and for $0 < \kappa \leq 1$, there exist $q_0 = q_0(\varepsilon) \in (0, \beta_0)$ and $\gamma_0 = \gamma_0(\kappa) > 0$, so that for all $\gamma > \gamma_0$*

$$\mathbb{P}\left(\sup_{R \in \mathcal{R}^{\kappa-reg};\, |R| \leq q_0} \frac{|B(R)|}{\sqrt{|R|}\sqrt{2\log\frac{1}{|R|} + \gamma \log\log\frac{1}{|R|}}} \leq 1 \right) \geq 1 - \varepsilon.$$

3. *For all $\varepsilon > 0$, there exists $\delta_0 = \delta_0(\varepsilon) > 2$, so that for all $\delta > \delta_0$*

$$\mathbb{P}\left(\sup_{R \in \mathcal{R};\, h_1, h_2 \leq \frac{1}{2}} \frac{|B(R)|}{\sqrt{|R|}\sqrt{\delta \log\frac{1}{|R|}}} \leq 1 \right) \geq 1 - \varepsilon.$$

4. *For all $\varepsilon > 0$ and for $0 < \kappa \leq 1$, there exists $\gamma_0 = \gamma_0(\varepsilon, \kappa) > 0$, so that for all $\gamma > \gamma_0$*

$$\mathbb{P}\left(\sup_{R \in \mathcal{R}^{\kappa-reg},\, |R| \neq 1} \frac{|B(R)|}{\sqrt{|R|}\sqrt{2\log\frac{1}{|R|} + \gamma \log\log\frac{1}{|R|}}} \leq 1 \right) \geq 1 - \varepsilon.$$

Parts 1 and 2 can be derived from more general propositions about the Brownian sheet, cf. e.g. Orey and Pruitt [OP73] or Alexander [Ale86]. Dümbgen and Spokoiny [DS01] consider specific stochastic processes on a pseudometric space with continuos sample paths and show a result corresponding to part 4.
Here we will give an elementary proof. For this purpose, we need one more lemma.

Lemma 5.7 *For $p \in \mathbb{N}$ and $m \in \mathbb{N}$ let $\mathcal{R}^p(m)$ be the set of all rectangles within the unit square of the form $[k_1 2^{-m_1}, (k_1 + l_1)2^{-m_1}] \times [k_2 2^{-m_2}, (k_2 + l_2)2^{-m_2}]$ with $m_1 + m_2 \geq m$, $1 \leq k_1 \leq 2^{m_1}$, $1 \leq l_1 \leq m_1^p$ and $1 \leq k_2 \leq 2^{m_2}$, $1 \leq l_2 \leq m_2^p$.*

1. *For all $\varepsilon > 0$, there exists $m_0 = m_0(\varepsilon) \in \mathbb{N}$ so that for all $\delta > 2$*

$$\mathbb{P}\left(\sup_{R \in \mathcal{R}^p(m_0)} \frac{|B(R)|}{\sqrt{|R|}\sqrt{\delta \log\frac{1}{|R|}}} \leq 1 \right) \geq 1 - \varepsilon.$$

2. *For all $\varepsilon > 0$, there exists $m_0 = m_0(\varepsilon) \in \mathbb{N}$ so that for all $\gamma > 8p + 3$*

$$\mathbb{P}\left(\sup_{R \in \mathcal{R}^p(m_0)} \frac{|B(R)|}{\sqrt{|R|}\sqrt{2\log\frac{1}{|R|} + \gamma \log\log\frac{1}{|R|}}} \leq 1 \right) \geq 1 - \varepsilon.$$

Proof
1. First of all we consider $\mathcal{R}^p(m_1,m_2) := \{R \in \mathcal{R};\ R = [k_1 2^{-m_1}, (k_1+l_1)2^{-m_1}] \times [k_2 2^{-m_2}, (k_2+l_2)2^{-m_2}],\ 1 \le k_1 \le 2^{m_1},\ 1 \le l_1 \le m_1^p,\ 1 \le k_2 \le 2^{m_2},\ 1 \le l_2 \le m_2^p\}$. Then $R \in \mathcal{R}^p(m_1,m_2)$ has the area $q := |R| = l_1 2^{-m_1} l_2 2^{-m_2} \le m_1^p\, 2^{-m_1} m_2^p\, 2^{-m_2}$. We have $\frac{|B(R)|}{\sqrt{q}} \sim \mathcal{N}(0,1)$, and thus with Lemma 5.1 we obtain

$$\mathbb{P}\left(\frac{|B(R)|}{\sqrt{q}} > \sqrt{\delta \log \frac{1}{q}}\right) \le \sqrt{\frac{2}{\pi}}\ \frac{1}{\sqrt{\delta \log \frac{1}{q}}}\ \exp\left(-\frac{\delta}{2}\log\frac{1}{q}\right)$$

$$\le C \frac{1}{\sqrt{m_1+m_2}}\ \left(2^{-m_1-m_2} m_1^p\, m_2^p\right)^{\frac{\delta}{2}} \le C\ \left(2^{-m_1-m_2}\right)^{\frac{\delta}{2}}\ (m_1+m_2)^{\delta p - \frac{1}{2}}$$

for large m_1+m_2 as $\log\frac{1}{q} = \log 2^{m_1+m_2} - l_1 \log m_1 - l_2 \log m_2 = O(m_1+m_2)$. Here $C>0$ is some constant.
With $\#\mathcal{R}^p(m_1,m_2) \le m_1^p m_2^p\, 2^{m_1} 2^{m_2} \le (m_1+m_2)^{2p}\, 2^{m_1+m_2}$, we obtain

$$\mathbb{P}\left(\sup_{R\in\mathcal{R}^p(m_1,m_2)} \frac{|B(R)|}{\sqrt{q}\sqrt{\delta\log\frac{1}{q}}} > 1\right) \le C\ \left(2^{-m_1-m_2}\right)^{\frac{\delta}{2}-1}\ (m_1+m_2)^{2p+\delta p-\frac{1}{2}}.$$

Now we want to consider $R \in \cup_{m_1,m_2}\mathcal{R}^p(m_1,m_2)$. We have

$$\mathbb{P}\left(\sup_{m_1,m_2}\ \sup_{R\in\mathcal{R}^p(m_1,m_2)} \frac{|B(R)|}{\sqrt{q}\sqrt{\delta\log\frac{1}{q}}} > 1\right) \le \sum_{m_1,\,m_2=1}^{\infty} \mathbb{P}\left(\sup_{R\in\mathcal{R}^p(m_1,m_2)} \frac{|B(R)|}{\sqrt{q}\sqrt{\delta\log\frac{1}{q}}} > 1\right)$$

$$\le C \sum_{m_1,\,m_2=1}^{\infty} (m_1+m_2)^{2p+\delta p-\frac{1}{2}}\ \left(2^{-m_1-m_2}\right)^{\frac{\delta}{2}-1} = C \sum_{m_1,\,m_2=1}^{\infty} (m_1+m_2)^{2p+\delta p-\frac{1}{2}}\ \left(2^{-\frac{\delta}{2}+1}\right)^{m_1+m_2}.$$

This sum is finite if $2^{-\frac{\delta}{2}+1} < 1$, which is equivalent to $-\frac{\delta}{2}+1<0$, i.e. to $\delta > 2$. Therefore for each $\varepsilon > 0$, there exists $m_0 = m_0(\varepsilon)$ so that for all $\delta > 2$

$$\sum_{m_1+m_2\ge m_0} (m_1+m_2)^{2p+\delta p-\frac{1}{2}}\ \left(2^{-m_1+m_2}\right)^{\frac{\delta}{2}-1} < \varepsilon.$$

This means that for $\mathcal{R}^p(m_0) = \cup_{m_1+m_2\ge m_0}\mathcal{R}^p(m_1,m_2)$, we have for all $\delta > 2$

$$\mathbb{P}\left(\sup_{R\in\mathcal{R}^p(m_0)} \frac{|B(R)|}{\sqrt{q}\sqrt{\delta\log\frac{1}{q}}} > 1\right) \le C \sum_{m_1+m_2\ge m_0} (m_1+m_2)^{2p+\delta p-\frac{1}{2}}\ \left(2^{-m_1+m_2}\right)^{\frac{\delta}{2}-1} < \varepsilon.$$

2. The structure of the proof of the second part is the same as the one of the first part. We start with $R \in \mathcal{R}^p(m_1, m_2)$. Lemma 5.1 gives us

$$\mathbb{P}\left(\frac{|B(R)|}{\sqrt{q}} > \sqrt{2\log\frac{1}{q} + \gamma \log\log\frac{1}{q}}\right)$$

$$\leq \sqrt{\frac{2}{\pi}}\, \frac{1}{\sqrt{2\log\frac{1}{q} + \gamma\log\log\frac{1}{q}}}\, \exp\left(-\log\frac{1}{q} - \frac{\gamma}{2}\log\log\frac{1}{q}\right)$$

$$\leq C\frac{1}{\sqrt{m_1+m_2}}\, 2^{-m_1-m_2} m_1^p\, m_2^p (m_1+m_2)^{-\frac{\gamma}{2}} \leq C\, 2^{-m_1-m_2}(m_1+m_2)^{2p-\frac{\gamma+1}{2}}$$

for large $m_1 + m_2$ and some constant $C > 0$. With $\#\mathcal{R}^p(m_1, m_2) \leq 2^{m_1}2^{m_2} \cdot (m_1+m_2)^{2p}$, we obtain

$$\mathbb{P}\left(\sup_{R\in\mathcal{R}^p(m_1,m_2)} \frac{|B(R)|}{\sqrt{q}\sqrt{2\log\frac{1}{q} + \gamma\log\log\frac{1}{q}}} > 1\right) \leq C\,(m_1+m_2)^{4p-\frac{\gamma+1}{2}}.$$

For $R \in \cup_{m_1,m_2}\mathbb{R}^p(m_1, m_2)$, we have

$$\mathbb{P}\left(\sup_{m_1,m_2}\ \sup_{R\in\mathcal{R}^p(m_1,m_2)} \frac{|B(R)|}{\sqrt{q}\sqrt{2\log\frac{1}{q} + \gamma\log\log\frac{1}{q}}} > 1\right)$$

$$\leq \sum_{m_1,\,m_2=1}^{\infty} \mathbb{P}\left(\sup_{R\in\mathcal{R}^p(m_1,m_2)} \frac{|B(R)|}{\sqrt{q}\sqrt{2\log\frac{1}{q} + \gamma\log\log\frac{1}{q}}} > 1\right) \leq \sum_{m_1,\,m_2=1}^{\infty} (m_1+m_2)^{4p-\frac{\gamma+1}{2}}.$$

This sum is finite, if $4p - \frac{\gamma+1}{2} < -2$, i.e. if $\gamma > 8p+3$. Therefore for each $\varepsilon > 0$, there exists $m_0 = m_0(\varepsilon)$ so that for all $\gamma > 8p+3$

$$\sum_{m_1+m_2\geq m_0} (m_1+m_2)^{4p-\frac{\gamma+1}{2}} < \varepsilon, \quad \text{which implies}$$

$$\mathbb{P}\left(\sup_{R\in\mathcal{R}^p(m_0)} \frac{|B(R)|}{\sqrt{q}\sqrt{2\log\frac{1}{q} + \gamma\log\log\frac{1}{q}}} > 1\right) < \varepsilon.$$

□

Proof of Theorem 5.6
We wish to show that the inequalities $B(R)| \leq \varphi_\delta(|R|)$ and $|B(R)| \leq \psi_\gamma(|R|)$ hold for all rectangles R, which are elements of different specific subsets of $\mathcal{R}$, simultaneously with a probability greater than $1-\varepsilon$. In order to show that the inequalities hold for general rectangles R of the subset, we approximate R by $R_m \in \mathcal{R}^p(m_1, m_2)$ for m_1, m_2 and p to be determined. We show that the error we make by this approximation is sufficiently small. When considering the supremum over all R, this supremum is reduced to the supremum over all R_m using the above approximation. Then we can use the statement of Lemma 5.7 for all the rectangles R_m.
Let $p \in \mathbb{N}$ be fixed. The choice of p will be specified later. For an arbitrary rectangle $R = [s_1, s_1 + h_1] \times [s_2, s_2 + h_2] \in \mathcal{R}$ we consider m_1 and m_2 with

$$(m_1+1)^p\, 2^{-(m_1+1)} < h_1 \leq m_1^p\, 2^{-m_1} \quad \text{and} \quad (m_2+1)^p\, 2^{-(m_2+1)} < h_2 \leq m_2^p\, 2^{-m_2}\,.$$

In the following we use the approximations $h_1 \approx m_1^p\, 2^{-m_1}$ and $h_2 \approx m_2^p\, 2^{-m_2}$.
Let $R_m \in \mathcal{R}^p(m_1, m_2)$ for $m = (m_1, m_2)$ be the maximal rectangle within R, and for each $j \in \mathbb{N}$ let $R_{m+j} \in \mathcal{R}^p(m_1 + j, m_2 + j)$ be maximal with $R_{m+j} \subseteq R$. Then

$$R_{m+j} \subseteq R_{m+j+1},$$

$$R = R_m \cup \overline{\bigcup_{j=0}^{\infty} R_{m+j+1} \setminus R_{m+j}} \quad \text{and}$$

$$|B(R)| \leq |B(R_m)| + \sum_{j=0}^{\infty} |B(R_{m+j+1} \setminus R_{m+j})|.$$

The area of the sets $R_{m+j+1} \setminus R_{m+j}$ can be approximated with $\eta_{i,j} \in \{0,1\}$ by

$$|R_{m+j+1} \setminus R_{m+j}| \approx \eta_{1,j} h_1 2^{-m_2-j} + \eta_{2,j} h_1 2^{-m_2-j} + \eta_{3,j} h_2 2^{-m_1-j} + \eta_{4,j} h_2 2^{-m_1-j}$$

$$\leq 2(h_1 2^{-m_2-j} + h_2 2^{-m_1-j}),$$

and thus $B(R_{m+j+1} \setminus R_{m+j}) \sim \mathcal{N}(0, \sigma^2)$ with

$$\sigma^2 \leq \sigma^2(m_1, m_2, j) = 2(h_1 2^{-m_2-j} + h_2 2^{-m_1-j})$$

$$\approx 2^{-j+1} \left(h_1 \frac{h_2}{m_2^p} + h_2 \frac{h_1}{m_1^p} \right) = h_1 h_2 2^{-j+1} \left(\frac{1}{m_1^p} + \frac{1}{m_2^p} \right)$$

$$\approx m_1^p\, 2^{-m_1}\, m_2^p\, 2^{-m_2}\, 2^{-j+1} \left(\frac{1}{m_1^p} + \frac{1}{m_2^p} \right)$$

$$= 2^{-m_1-m_2}\, 2^{-j+1}\, (m_1^p + m_2^p) \leq 2^{-j+2}\, 2^{-m_1-m_2}\, (m_1+m_2)^p\,.$$

For large m_1 and m_2, we have the approximation

$$\log \frac{1}{\sigma^2(m_1, m_2, j)} \approx \log \frac{1}{h_1 h_2} \approx m_1 + m_2.$$

For proving the first statement of the theorem, we approximate the rectangle $R \in \mathcal{R}$ by R_m and show that the error we make by doing so is small. $q_0 = q_0(\varepsilon)$ is chosen such that the resulting $m_1 + m_2 \geq m_0 = m_0(\varepsilon)$ from Lemma 5.7. Thus for R with $|R| \leq q_0$, the resulting R_m is an element of $\mathcal{R}^p(m_0)$, and we know from Lemma 5.7 that for all $\delta > 2$

$$\mathbb{P}\left(\sup_{R_m \in \mathcal{R}^p(m_0)} \frac{|B(R_m)|}{\sqrt{|R_m|}\sqrt{\delta \frac{1}{|R_m|}}} \leq 1 \right) \geq 1 - \varepsilon.$$

The value of q_0 might be lowered during the proof.
In order to estimate the error we make by approximating R by R_m, we consider the sets $R_{m+j+1} \setminus R_{m+j}$. From Lemma 5.1 we get

$$\mathbb{P}\left(\frac{|B(R_{m+j+1} \setminus R_{m+j})|}{\sigma(m_1, m_2, j)} > \sqrt{\delta \log \frac{1}{\sigma^2(m_1, m_2, j)}} \right)$$

$$\leq 2 \exp\left(- \frac{\delta \log \frac{1}{\sigma^2(m_1, m_2, j)}}{2} \right) = 2(\sigma^2(m_1, m_2, j))^{\frac{\delta}{2}}$$

$$\approx 2\left(2^{-m_1 - m_2}\, 2^{-j+1}\, (m_1^p + m_2^p)\right)^{\frac{\delta}{2}}.$$

Regarding all the sets $R_{m+j+1} \setminus R_{m+j}$ for $j = 0, 1, \ldots$ at the same time, we obtain

$$\mathbb{P}\left(\sup_j \frac{|B(R_{m+j+1} \setminus R_{m+j})|}{\sigma(m_1, m_2, j)\sqrt{\delta \log \frac{1}{\sigma^2(m_1, m_2, j)}}} > 1 \right)$$

$$\leq \sum_{j=0}^{\infty} 2\left(2^{-m_1 - m_2}\, 2^{-j+1}\, (m_1^p + m_2^p)\right)^{\frac{\delta}{2}} \leq C \cdot \left(2^{-m_1 - m_2}\right)^{\frac{\delta}{2}}\, (m_1 + m_2)^{p\frac{\delta}{2}}$$

for some constant $C > 0$.

Now we consider all the rectangles $R \in \mathcal{R}$ whose side lengths h_1 and h_2 yield the same pair (m_1, m_2). We denote this set by $\mathcal{R}^{p\star}(m_1, m_2)$. Then for all $R \in \mathcal{R}^{p\star}(m_1, m_2)$, we have at most $\#\mathcal{R}^p(m_1, m_2) \leq m_1^p\, 2^{m_1}\, m_2^p\, 2^{m_2} \leq (m_1 + m_2)^{2p}\, 2^{m_1 + m_2}$ possible rectangles R_m corresponding to R, and

$$\mathbb{P}\left(\sup_{R\in\mathcal{R}^{p\star}(m_1,m_2)}\ \sup_j\ \frac{|B(R_{m+j+1}\setminus R_{m+j})|}{\sigma(m_1,m_2,j)\sqrt{\delta\log\frac{1}{\sigma^2(m_1,m_2,j)}}}>1\right)$$
$$\leq C\ \left(2^{m_1+m_2}\right)^{1-\frac{\delta}{2}}\ (m_1+m_2)^{p\frac{\delta}{2}+2p}.$$

For the supremum over all $R\in\cup_{m_1,m_2}\mathcal{R}^{p\star}(m_1,m_2)=\mathcal{R}$, we have

$$\mathbb{P}\left(\sup_{m_1,m_2}\ \sup_{R\in\mathcal{R}^{p\star}(m_1,m_2)}\ \sup_j\ \frac{|B(R_{m+j+1}\setminus R_{m+j})|}{\sigma(m_1,m_2,j)\sqrt{\delta\log\frac{1}{\sigma^2(m_1,m_2,j)}}}>1\right)$$

$$\leq C\sum_{m_1,m_2=1}^{\infty}\left(2^{m_1+m_2}\right)^{1-\frac{\delta}{2}}\ (m_1+m_2)^{p\frac{\delta}{2}+2p}=C\sum_{m_1,m_2=1}^{\infty}\left(2^{1-\frac{\delta}{2}}\right)^{m_1+m_2}\ (m_1+m_2)^{p\frac{\delta}{2}+2p}.$$

This sum is finite, if $2^{1-\frac{\delta}{2}}<1$, which is fulfilled for $1-\frac{\delta}{2}<0$, i.e. for $\delta>2$. Hence for each $\varepsilon>0$, there exists $m_0=m_0(\varepsilon)$ so that for all $\delta>0$

$$C\sum_{m_1+m_2\geq m_0=m_0(\varepsilon)}\left(2^{m_1+m_2}\right)^{1-\frac{\delta}{2}}\ (m_1+m_2)^{p\frac{\delta}{2}+2p}<\varepsilon,$$

which implies for $\mathcal{R}^{p\star}(m_0):=\cup_{m_1+m_2\geq m_0}\mathcal{R}^{p\star}(m_1,m_2)$ that

$$\mathbb{P}\left(\sup_{R\in R^{p\star}(m_0)}\ \sup_j\ \frac{|B(R_{m+j+1}\setminus R_{m+j})|}{\sigma(m_1,m_2,j)\sqrt{\delta\log\frac{1}{\sigma^2(m_1,m_2,j)}}}>1\right)\leq\varepsilon.$$

The set $\mathcal{R}^{p\star}(m_0)$ consists of all the rectangles $R\in\mathcal{R}$ whose side lengths h_1 and h_2 yield (m_1,m_2) with $m_1+m_2\geq m_0=m_0(\varepsilon)$. The condition $m_1+m_2\geq m_0$ corresponds via $\log\frac{1}{h_i}\approx m_i$ to the condition $h_1h_2\leq q_0=q_0(\varepsilon)$.

In the following, we estimate the error we make by approximating the rectangle R by R_m. Since

$$R\setminus R_m=\overline{\bigcup_{j=0}^{\infty}R_{m+j+1}\setminus R_{m+j}},$$

we have

$$|B(R\setminus R_m)|\leq\sum_{j=0}^{\infty}|B(R_{m+j+1}\setminus R_{m+j})|.$$

If $|R| = h_1 h_2 \le q_0(\varepsilon)$, so that the corresponding $m_1 + m_2 \ge m_0(\varepsilon)$, we know with a probability greater than $1 - \varepsilon$ that we have

$$|B(R_{m+j+1} \setminus R_{m+j})| \le \sigma(m_1, m_2, j) \sqrt{\delta' \log \frac{1}{\sigma^2(m_1, m_2, j)}}$$

$$\approx \sqrt{h_1 h_2 2^{-j} \left(\frac{1}{m_1^p} + \frac{1}{m_2^p} \right)} \sqrt{\delta' \log \frac{1}{h_1 h_2}}$$

for all j and small h_1, h_2. Hence we obtain for $h_1 h_2 \le q_0$ with a probability greater than $1 - \varepsilon$ that

$$|B(R \setminus R_m)| \le \sum_{j=0}^{\infty} |B(R_{m+j+1} \setminus R_{m+j})|$$

$$\le \sum_{j=0}^{\infty} \sqrt{h_1 h_2 2^{-j} \left(\frac{1}{m_1^p} + \frac{1}{m_2^p} \right)} \sqrt{\delta' \log \frac{1}{h_1 h_2}} = 2\sqrt{\delta'} \sqrt{h_1 h_2 \left(\frac{1}{m_1^p} + \frac{1}{m_2^p} \right)} \sqrt{\log \frac{1}{h_1 h_2}}.$$

For the rectangle R_m, we obtain by Lemmas 5.2 and 5.7 that with a probability greater than $1 - \varepsilon$

$$|B(R_m)| \le \varphi_{\delta'}(|R_m|) \le \varphi_{\delta'}(|R|) = \varphi_{\delta'}(h_1 h_2) = \sqrt{h_1 h_2} \sqrt{\delta' \log \frac{1}{h_1 h_2}} \quad \forall \delta' > 2.$$

Recall that our aim is to show that the inequality

$$|B(R)| \le \varphi_{\delta}(|R|) = \sqrt{h_1 h_2} \sqrt{\delta \log \frac{1}{h_1 h_2}} \qquad \forall \delta > 2$$

holds for all $R \in \mathcal{R}$ with $h_1, h_2 \le \frac{1}{2}$ and $|R| \le q_0$ simultaneously with a probability greater than $1 - \varepsilon$.
We set $\xi := \delta - 2$. As the inequality can be shown easily for large δ, we now consider those $\delta > 2$ which are close to 2, so the corresponding ξ is close to 0. Using the approximation

$$\sqrt{2 + \xi} = \sqrt{2} \sqrt{1 + \frac{\xi}{2}} \approx \sqrt{2} \left(1 + \frac{\xi}{2 \cdot 2} \right) = \sqrt{2} + \frac{\xi}{2\sqrt{2}} \quad \text{for } \xi << 2,$$

we obtain

$$|B(R)| \le \sqrt{h_1 h_2} \sqrt{\log \frac{1}{h_1 h_2}} \left(\sqrt{2} + \frac{\xi}{2\sqrt{2}} \right)$$

as a new inequality to be shown.
The same approximation for R_m with $\delta' - 2 =: \xi' << 2$ yields

$$|B(R_m)| \le \sqrt{h_1 h_2} \sqrt{\log \frac{1}{h_1 h_2}} \left(\sqrt{2} + \frac{\xi'}{2\sqrt{2}} \right)$$

with a probability greater than $1-\varepsilon$.
Now we use $|B(R)| \leq |B(R_m)| + \sum_{j=0}^{\infty} |B(R_{m+j+1} \setminus R_{m+j})|$, and obtain

$$|B(R)| \leq \sqrt{h_1 h_2}\sqrt{\log \frac{1}{h_1 h_2}} \left(\sqrt{2} + \frac{\xi'}{2\sqrt{2}} + 2\sqrt{2+\xi'}\sqrt{\frac{1}{m_1^p} + \frac{1}{m_2^p}} \right)$$

with a probability greater than $1-\varepsilon$. If

$$2\sqrt{2+\xi'}\sqrt{\frac{1}{m_1^p} + \frac{1}{m_2^p}} \leq \frac{\xi'}{2\sqrt{2}}$$

holds, we have

$$|B(R)| \leq \sqrt{h_1 h_2}\sqrt{\log \frac{1}{h_1 h_2}} \left(\sqrt{2} + 2\frac{\xi'}{2\sqrt{2}} \right),$$

and for each $\xi > 0$, we can use $\xi' := \frac{\xi}{2} > 0$ to attain our aim

$$|B(R)| \leq \sqrt{h_1 h_2}\sqrt{\log \frac{1}{h_1 h_2}} \left(\sqrt{2} + \frac{\xi}{2\sqrt{2}} \right) \approx \varphi_\delta(|R|)$$

with a probability greater than $1-\varepsilon$. Hence, we need to show that

$$2\sqrt{2+\xi'}\sqrt{\frac{1}{m_1^p} + \frac{1}{m_2^p}} \leq \frac{\xi'}{2\sqrt{2}},$$

which is equivalent to

$$\frac{1}{m_1^p} + \frac{1}{m_2^p} \leq \frac{\xi'^2}{32(2+\xi')}\,.$$

$h_1 \leq \frac{1}{2}$ and $h_2 \leq \frac{1}{2}$ correspond to $m_1 \geq 2$ and $m_2 \geq 2$. Therefore

$$\frac{1}{m_1^p} + \frac{1}{m_2^p} \leq \frac{2}{2^p},$$

and for each $\xi > 0$, i.e. for each $\delta > 2$, we can find $p \in \mathbb{N}$, so that

$$\frac{2}{2^p} \leq \frac{\xi'^2}{32(2+\xi')}\,.$$

This shows that $|B(R)| \leq \varphi_\delta(|R|)$ with a probability greater than $1-\varepsilon$ for all $\delta > 2$ and the chosen $p \in \mathbb{N}$. Note that p only depends on δ.

It remains to show that the inequality holds for the supremum over all $R \in \mathcal{R}$ with $h_1, h_2 \leq \frac{1}{2}$ and $|R| \leq q_0$. This supremum reduces to the supremum over all $R_m \in \mathcal{R}^p(m_0)$, for which we can use Lemma 5.7. As

$$\sup_{R\in\mathcal{R}:\, h_1,h_2\leq\frac{1}{2},\,|R|\leq q_0} \frac{|B(R)|}{\sqrt{|R|}\sqrt{\delta\log\frac{1}{|R|}}} \leq \sup_{R\in\mathcal{R}:\, h_1,h_2\leq\frac{1}{2},\,|R|\leq q_0} \frac{|B(R_m)|+|B(R\setminus R_m)|}{\sqrt{|R|}\sqrt{\delta\log\frac{1}{|R|}}}$$

$$\leq \sup_{R_m\in\mathcal{R}^p(m_0)} \frac{|B(R_m)|+\sum_{j=0}^{\infty}|B(R_{m+j+1}\setminus R_{m+j})|}{\sqrt{|R_m|}\sqrt{\delta\log\frac{1}{|R_m|}}},$$

we have

$$\mathbb{P}\left(\sup_{R\in\mathcal{R};\, h_1,h_2\leq\frac{1}{2},\,|R|\leq q_0} \frac{|B(R)|}{\sqrt{|R|}\sqrt{\delta\log\frac{1}{|R|}}} > 1\right)$$

$$\leq \mathbb{P}\left(\sup_{R_m\in\mathcal{R}^p(m_0)} \frac{|B(R_m)|+\sum_{j=0}^{\infty}|B(R_{m+j+1}\setminus R_{m+j})|}{\sqrt{|R_m|}\sqrt{\delta\log\frac{1}{|R_m|}}} > 1\right)$$

$$\leq \mathbb{P}\left(\sup_{R_m\in\mathcal{R}^p(m_0)} \frac{|B(R_m)|}{\sqrt{|R_m|}\sqrt{\delta\log\frac{1}{|R_m|}}} > 1\right) \leq \varepsilon.$$

This proves the first part of the theorem.

The second part is shown in an analogous way. The computations are mostly similar, therefore we only state the main results. We approximate R by R_m in the same way as before. For R with $|R| \leq q_0(\varepsilon)$ such that the corresponding $R_m \in \mathcal{R}^p(m_0)$, we know from Lemma 5.7 that for all $\gamma' > 8p+3$

$$\mathbb{P}\left(\sup_{R_m\in\mathcal{R}^p(m_0)} \frac{|B(R_m)|}{\sqrt{|R_m|}\sqrt{2\log\frac{1}{|R_m|}+\gamma'\log\log\frac{1}{|R_m|}}} \leq 1\right) \geq 1-\varepsilon.$$

In order to estimate the error we make by approximating R by R_m, we first consider the sets $R_{m+j+1}\setminus R_{m+j}$. We use Lemma 5.1 to obtain

$$\mathbb{P}\left(\frac{|B(R_{m+j+1}\setminus R_{m+j})|}{\sigma(m_1,m_2,j)} > \sqrt{2\log\frac{1}{\sigma^2(m_1,m_2,j)}+\gamma'\log\log\frac{1}{\sigma^2(m_1,m_2,j)}}\right)$$

$$\leq 2\exp\left(-\frac{2\log\frac{1}{\sigma^2(m_1,m_2,j)}+\gamma'\log\log\frac{1}{\sigma^2(m_1,m_2,j)}}{2}\right)$$

$$= 2\,\sigma^2(m_1,m_2,j)\,\left(\log\frac{1}{\sigma^2(m_1,m_2,j)}\right)^{-\frac{1+\gamma'}{2}}$$

$$\leq 2\cdot 2^{-j+2}2^{-m_1-m_2}(m_1+m_2)^p\cdot(m_1+m_2)^{-\frac{1+\gamma'}{2}}$$

$$= 2^{-j+3}2^{-m_1-m_2}\,(m_1+m_2)^{p-\frac{1+\gamma'}{2}}.$$

For the supremum over all the sets $R_{m+j+1}\setminus R_{m+j}$ for $j=0,1,\ldots$ we get

$$\mathbb{P}\left(\sup_j \frac{|B(R_{m+j+1}\setminus R_{m+j})|}{\sigma(m_1,m_2,j)\sqrt{2\log\frac{1}{\sigma^2(m_1,m_2,j)}+\gamma'\log\log\frac{1}{\sigma^2(m_1,m_2,j)}}}>1\right)$$

$$\leq C\cdot 2^{-m_1-m_2}\,(m_1+m_2)^{p-\frac{1+\gamma'}{2}}\quad\text{for some constant } C>0.$$

Now we consider all the rectangles $R\in\mathcal{R}$ which yield the same pair (m_1,m_2), i.e. $R\in\mathcal{R}^{p\star}(m_1,m_2)$. With $\#\mathcal{R}^p(m_1,m_2)\leq(m_1+m_2)^{2p}\,2^{m_1+m_2}$, we obtain

$$\mathbb{P}\left(\sup_{R\in\mathcal{R}^{p\star}(m_1,m_2)}\sup_j \frac{|B(R_{m+j+1}\setminus R_{m+j})|}{\sigma(m_1,m_2,j)\sqrt{2\log\frac{1}{\sigma^2(m_1,m_2,j)}+\gamma'\log\log\frac{1}{\sigma^2(m_1,m_2,j)}}}>1\right)$$

$$\leq C\,(m_1+m_2)^{3p-\frac{1+\gamma'}{2}}.$$

For the supremum over all $R\in\cup_{m_1,m_2}\mathcal{R}^{p\star}(m_1,m_2)=\mathcal{R}$ we have

$$\mathbb{P}\left(\sup_{m_1,m_2}\sup_{R\in\mathcal{R}^{p\star}(m_1,m_2)}\sup_j \frac{|B(R_{m+j+1}\setminus R_{m+j})|}{\sigma(m_1,m_2,j)\sqrt{2\log\frac{1}{\sigma^2(m_1,m_2,j)}+\gamma'\log\log\frac{1}{\sigma^2(m_1,m_2,j)}}}>1\right)$$

$$\leq C\sum_{m_1,m_2=1}^{\infty}(m_1+m_2)^{3p-\frac{1+\gamma'}{2}}.$$

This sum is finite, if $3p-\frac{1+\gamma'}{2}<-2$, i.e. if $\gamma'>3+6p$. Hence for each $\varepsilon>0$, there exists $m_0=m_0(\varepsilon)$ so that for all $\gamma'>3+6p$

$$C\sum_{m_1+m_2\geq m_0=m_0(\varepsilon)}(m_1+m_2)^{3p-\frac{1+\gamma'}{2}}<\varepsilon,$$

which implies for $\mathcal{R}^{p\star}(m_0):=\cup_{m_1+m_2\geq m_0}\mathcal{R}^{p\star}(m_1,m_2)$ that

$$\mathbb{P}\left(\sup_{R\in R^{p\star}(m_0)}\sup_j \frac{|B(R_{m+j+1}\setminus R_{m+j})|}{\sigma(m_1,m_2,j)\sqrt{2\log\frac{1}{\sigma^2(m_1,m_2,j)}+\gamma'\log\log\frac{1}{\sigma^2(m_1,m_2,j)}}}>1\right)\leq\varepsilon.$$

Therefore if $|R| \leq q_0(\varepsilon)$, we have with a probability greater than $1-\varepsilon$

$$|B(R_{m+j+1} \setminus R_{m+j})| \leq \sigma(m_1, m_2, j)\sqrt{2\log\frac{1}{\sigma^2(m_1, m_2, j)} + \gamma' \log\log\frac{1}{\sigma^2(m_1, m_2, j)}}$$

$$\approx \sqrt{h_1 h_2 2^{-j}\left(\frac{1}{m_1^p} + \frac{1}{m_2^p}\right)}\sqrt{2\log\frac{1}{h_1 h_2} + \gamma' \log\log\frac{1}{h_1 h_2}}$$

$$\approx \sqrt{h_1 h_2 2^{-j}\left(\frac{1}{m_1^p} + \frac{1}{m_2^p}\right)}\sqrt{2\log\frac{1}{h_1 h_2}} \quad \text{for all } j,$$

since $\gamma' \log\log\frac{1}{h_1 h_2} = o\left(\log\frac{1}{h_1 h_2}\right)$ as $h_1 h_2 \to 0$. Hence for the error of the approximation, we obtain with a probability greater than $1-\varepsilon$

$$|B(R \setminus R_m)| \leq \sum_{j=0}^{\infty} |B(R_{m+j+1} \setminus R_{m+j})|$$

$$\leq \sum_{j=0}^{\infty} \sqrt{2^{-j}}\sqrt{h_1 h_2\left(\frac{1}{m_1^p} + \frac{1}{m_2^p}\right)}\sqrt{2\log\frac{1}{h_1 h_2}}$$

$$\leq C\sqrt{h_1 h_2\left(\frac{1}{m_1^p} + \frac{1}{m_2^p}\right)}\sqrt{\log\frac{1}{h_1 h_2}} \quad \text{for some constant } C > 0.$$

Our aim is to show that we can find γ, such that

$$|B(R)| \leq \psi_\gamma(|R|) = \sqrt{h_1 h_2}\sqrt{2\log\frac{1}{h_1 h_2} + \gamma \log\log\frac{1}{h_1 h_2}}$$

for all $R \in \mathcal{R}^{\kappa-reg}$ with $|R| \leq q_0$ with a probability greater than $1-\varepsilon$.
Using for $2\log\frac{1}{h_1 h_2} =: a >> b := \gamma \log\log\frac{1}{h_1 h_2}$ the approximation

$$\sqrt{a+b} \approx \sqrt{a} + \frac{b}{2\sqrt{a}},$$

this can be written as

$$|B(R)| \leq \sqrt{h_1 h_2}\left(\sqrt{2\log\frac{1}{h_1 h_2}} + \gamma\frac{\log\log\frac{1}{h_1 h_2}}{2\sqrt{2\log\frac{1}{h_1 h_2}}}\right).$$

The same approximation can be done for R_m, and we get

$$|B(R_m)| \leq \psi_{\gamma'}(|R_m|) \leq \psi_{\gamma'}(|R|)$$

$$\approx \sqrt{h_1 h_2}\left(\sqrt{2\log\frac{1}{h_1h_2}}+\gamma'\frac{\log\log\frac{1}{h_1h_2}}{2\sqrt{2\log\frac{1}{h_1h_2}}}\right)$$

with a probability greater than $1-\varepsilon$. With $|B(R)| \le |B(R_m)| + |B(R\setminus R_m)|$, this yields

$$|B(R)| \le \sqrt{h_1 h_2}\left(\sqrt{2\log\frac{1}{h_1h_2}}+\gamma'\frac{\log\log\frac{1}{h_1h_2}}{2\sqrt{2\log\frac{1}{h_1h_2}}}+C\sqrt{\left(\frac{1}{m_1^p}+\frac{1}{m_2^p}\right)}\sqrt{\log\frac{1}{h_1h_2}}\right)$$

with a probability greater than $1-\varepsilon$. If we can show that

$$C\sqrt{\left(\frac{1}{m_1^p}+\frac{1}{m_2^p}\right)}\sqrt{\log\frac{1}{h_1h_2}} \le \frac{\log\log\frac{1}{h_1h_2}}{2\sqrt{2\log\frac{1}{h_1h_2}}},$$

it follows that with a probability greater than $1-\varepsilon$

$$|B(R)| \le \sqrt{h_1 h_2}\left(\sqrt{2\log\frac{1}{h_1h_2}}+(\gamma'+1)\frac{\log\log\frac{1}{h_1h_2}}{2\sqrt{2\log\frac{1}{h_1h_2}}}\right) \approx \psi_\gamma(h_1h_2)$$

with $\gamma := \gamma' + 1$.
With $\log\frac{1}{h_1h_2} = O(m_1+m_2)$ and $\log\log\frac{1}{h_1h_2} = O(\log(m_1+m_2))$ as $m_1+m_2\to\infty$, this inequality results for small h_1h_2 in

$$\sqrt{\frac{1}{m_1^p}+\frac{1}{m_2^p}} \le C'\frac{\log(m_1+m_2)}{m_1+m_2} \quad \text{for some constant } C'>0.$$

At first we consider the left side $\sqrt{\frac{1}{m_1^p}+\frac{1}{m_2^p}}$ of the inequality. Let R be a κ-regular rectangle, and without loss of generality let $h_1 \le h_2$. Then we have

$$\log\frac{1}{h_1} \ge \log\frac{1}{h_2} \ge \left(\log\frac{1}{h_1}\right)^\kappa,$$

which yields $m_1 \ge m_2 \ge m_1^\kappa$ and thus $\frac{1}{m_2^p} \le \frac{1}{m_1^{\kappa p}}$ and $\frac{1}{m_1^p} \le \frac{1}{m_1^{\kappa p}}$. Hence we obtain

$$\frac{1}{m_2^p}+\frac{1}{m_1^p} \le \frac{2}{m_1^{\kappa p}}.$$

Altogether we have for the left side

$$\sqrt{\frac{1}{m_2^p}+\frac{1}{m_1^p}} \le \sqrt{\frac{2}{m_1^{\kappa p}}} = C_1\, m_1^{-\frac{\kappa p}{2}} \quad \text{for some constant } C_1>0.$$

Now we consider the right side $C' \frac{\log(m_1+m_2)}{m_1+m_2}$ of the ineqality. As $h_1 \leq h_2$, we have $m_1 \geq m_2$, and hence

$$\frac{\log(m_1+m_2)}{m_1+m_2} \geq \frac{\log(2m_1)}{2m_1} = O(m_1^{-1}) \qquad \text{as } m_1 \to \infty.$$

Thus we have for the right side

$$C' \frac{\log(m_1+m_2)}{m_1+m_2} \geq C_2\, m_1^{-1} \quad \text{for some constant } C_2 > 0.$$

So we have to show that $m_1^{-\frac{\kappa p}{2}} \leq C_3\, m_1^{-1}$ for some constant $C_3 > 0$. For the sake of simplicity we set $C_3 = 1$. The inequality $m_1^{-\frac{\kappa p}{2}} \leq m_1^{-1}$ is satisfied if $\kappa p \geq 2$. So for each fixed κ we can find $p = p(\kappa) \geq \frac{2}{\kappa}$, such that for each $\varepsilon > 0$ and each $\gamma = \gamma' + 1 > 8p(\kappa) + 4 =: \gamma_0(\kappa)$, we obtain

$$\mathbb{P}\left(\sup_{R \in \mathcal{R}^{\kappa-reg},\, |R| \leq q_0} \frac{|B(R)|}{\psi_\gamma(|R|)} > 1 \right) \leq \mathbb{P}\left(\sup_{R_m \in \mathcal{R}^p(m_0) \cap \mathcal{R}^{\kappa-reg}} \frac{|B(R_m)|}{\psi_{\gamma'}(|R_m|)} > 1 \right) \leq \varepsilon.$$

Note that $C_3 \neq 1$ only changes the value of $p(\kappa)$.
This proves the second part of the theorem.

Proving the last two parts of the theorem, we use the arguments above with $\frac{\varepsilon}{2}$. We need to show that

$$\mathbb{P}\left(\sup_{R \in \mathcal{R},\, |R| > q_0} \frac{|B(R)|}{\sqrt{|R|}\sqrt{\delta \log \frac{1}{|R|}}} \leq 1 \right) \geq 1 - \frac{\varepsilon}{2}, \quad \text{for part 3, and}$$

$$\mathbb{P}\left(\sup_{R \in \mathcal{R}^{\kappa-reg},\, |R| > q_0} \frac{|B(R)|}{\sqrt{|R|}\sqrt{2\log \frac{1}{|R|} + \gamma \log\log \frac{1}{|R|}}} \leq 1 \right) \geq 1 - \frac{\varepsilon}{2},$$

for part 4. $|R| > q_0$ corresponds to $m_1 + m_2 < m_0$ for the approximating rectangle R_m. Hence there are only finitely many approximating rectangles, and $\gamma(\varepsilon, \kappa)$ and $\delta(\varepsilon)$ can be chosen such that the inequalities are fulfilled (cf. the proof of Theorem 5.3).

□

5.2.2 The Modulus of Continuity for Wedges

Now we wish to show a corresponding result for wedges. A wedge is defined by its up to five vertices.
First of all we need to define $B(W)$ for a wedge W in analogy to $B(R)$ for a rectangle R. The following definition does so for general regions A within the unit square.

Definition 5.8 *Given the i.i.d. random variables $Z_{ij} \sim \mathcal{N}(0,1)$ for $0 \leq i \leq m$, $0 \leq j \leq n$ and $N = (m+1)\cdot(n+1)$, as well as a region $A \subseteq [0,1]^2$, we define*

1. *the random variables $B_{m,n}(A)$ by*

$$B_{m,n}(A) = \frac{1}{\sqrt{N}} \sum_{i,j:\,(\frac{i}{m},\frac{j}{n})\in A} Z_{ij},$$

and a norm $|A|_{m,n}$ by

$$|A|_{m,n} = \frac{\#\{(i,j) : (\frac{i}{m},\frac{j}{n}) \in A\}}{N},$$

2. *the random variables $B(A)$ by*

$$B(A) = \lim_{m,n\to\infty} B_{m,n}(A)$$

and a norm $|A|$ by

$$|A| = \lim_{m,n\to\infty} |A|_{m,n}.$$

Then $B(A) \sim \mathcal{N}(0,|A|)$ are normally distributed random variables, which are independent for disjoint regions A, and $|A|$ is the same as the area of A.
Considering all the rectangles of the form $[0,s_1]\times[0,s_2]$ with $s_1, s_2 \in [0,1]$, the random variables $B([0,s_1]\times[0,s_2])$ define a Brownian sheet.

Now we proceed to the main theorem about wedges.

Theorem 5.9

1. *For all $\varepsilon > 0$, there exists $q_0 = q_0(\varepsilon) \in (0,\alpha_0)$, so that for all $\delta > 2$*

$$\mathbb{P}\left(\sup_{W\in\mathcal{W};\, h_1,h_2\leq\frac{1}{2},\, |R(W)|\leq q_0} \frac{|B(W)|}{\sqrt{|W|}\sqrt{\delta\log\frac{1}{|W|}}} \leq 1\right) \geq 1-\varepsilon.$$

2. *For all* $\varepsilon > 0$ *and for* $0 < \kappa \leq 1$, *there exist* $q_0 = q_0(\varepsilon) \in (0, \beta_0)$ *and* $\gamma_0 = \gamma_0(\kappa) > 0$, *so that for all* $\gamma > \gamma_0$

$$\mathbb{P}\left(\sup_{W \in \mathcal{W}^{\kappa-reg};\, |R(W)| \leq q_0} \frac{|B(W)|}{\sqrt{|W|}\sqrt{2\log\frac{1}{|W|} + \gamma \log\log\frac{1}{|W|}}} \leq 1 \right) \geq 1 - \varepsilon.$$

3. *For all* $\varepsilon > 0$, *there exists* $\delta_0 = \delta_0(\varepsilon) > 0$, *so that for all* $\delta > \delta_0$

$$\mathbb{P}\left(\sup_{W \in \mathcal{W};\, h_1, h_2 \leq \frac{1}{2}} \frac{|B(W)|}{\sqrt{|W|}\sqrt{\delta \log\frac{1}{|W|}}} \leq 1 \right) \geq 1 - \varepsilon.$$

4. *For all* $\varepsilon > 0$ *and for* $0 < \kappa \leq 1$, *there exists* $\gamma_0 = \gamma_0(\varepsilon, \kappa) > 0$, *so that for all* $\gamma > \gamma_0$

$$\mathbb{P}\left(\sup_{W \in \mathcal{W}^{\kappa-reg},\, |W| \neq 1} \frac{|B(W)|}{\sqrt{|W|}\sqrt{2\log\frac{1}{|W|} + \gamma \log\log\frac{1}{|W|}}} \leq 1 \right) \geq 1 - \varepsilon.$$

We remark that in analogy with Theorem 5.6 for rectangles, parts 1 and 2 can be derived from the main result of Alexander [Ale86]. Again we prefer an elementary proof. Since it is similar to that of Theorem 5.6, we do not repeat all the computations. In addition, we restrict ourselves to prove only part 2 of the theorem. We start with an analogue of Lemma 5.7.

Lemma 5.10 *For* $p \in \mathbb{N}$ *let* $\mathcal{W}^p(m)$ *be the set of all wedges* W, *whose defining vertices are of the form* $(k_1\, 2^{-m_1},\, k_2\, 2^{-m_2})$ *with* $k_1, k_2 \in \mathbb{N}$ *and* $m_1 + m_2 \geq m$, *and for which the corresponding rectangle* $R(W)$ *is an element of* $\mathcal{R}^p(m)$.

1. *For all* $\varepsilon > 0$, *there exists* $m_0 = m_0(\varepsilon) \in \mathbb{N}$ *so that for all* $\delta > 2$

$$\mathbb{P}\left(\sup_{W \in \mathcal{W}^p(m_0)} \frac{|B(W)|}{\sqrt{|W|}\sqrt{\delta \log\frac{1}{|W|}}} \leq 1 \right) \geq 1 - \varepsilon.$$

2. *For all* $\varepsilon > 0$, *there exists* $m_0 = m_0(\varepsilon) \in \mathbb{N}$ *so that for all* $\gamma > 12p + 3$

$$\mathbb{P}\left(\sup_{W \in \mathcal{W}^p(m_0)} \frac{|B(W)|}{\sqrt{|W|}\sqrt{2\log\frac{1}{|W|} + \gamma \log\log\frac{1}{|W|}}} \leq 1 \right) \geq 1 - \varepsilon.$$

Proof
We restrict to the proof of part 2 and start considering the set $\mathcal{W}^p(m_1, m_2) := \{W \in \mathcal{W} : \text{each vertex of } W \text{ is of the form } (k_1 2^{-m_1}, k_2 2^{-m_2}),\ k_1, k_2 \in \mathbb{N} \text{ and } R(W) \in \mathcal{R}^p(m_1, m_2)\}$.
Let $W \in \mathcal{W}^p(m_1, m_2)$. Then we obtain for the area of W

$$w := |W| \leq |R(W)| = l_1 l_2\, 2^{-m_1-m_2} \leq m_1^p\, m_2^p\, 2^{-m_1-m_2} \quad \text{and}$$

$$w \geq \frac{1}{2}\,|R(W)| \geq 2^{-m_1-m_2-1}.$$

Therefore we have $\log \frac{1}{w} \approx m_1 + m_2$ and $\sqrt{2\log\frac{1}{w} + \gamma \log\log\frac{1}{w}} \approx \sqrt{2(m_1+m_2)}$ for large $m_1 + m_2$. Thus we obtain from Lemma 5.1

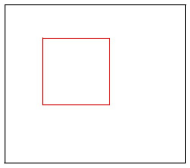
$$\mathbb{P}\left(\frac{|B(W)|}{\sqrt{w}} > \sqrt{2\log\frac{1}{w} + \gamma\log\log\frac{1}{w}}\right)$$

$$\leq \frac{1}{\sqrt{2\log\frac{1}{w} + \gamma\log\log\frac{1}{w}}}\ \exp\left(-\log\frac{1}{w} - \frac{\gamma}{2}\log\log\frac{1}{w}\right) \approx \frac{1}{\sqrt{2(m_1+m_2)}}\, w\ (\log w)^{\frac{\gamma}{2}}$$

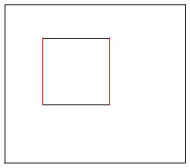
$$\leq \frac{1}{\sqrt{m_1+m_2}}\, m_1^p\, m_2^p\, 2^{-m_1-m_2}\, (m_1+m_2)^{-\frac{\gamma}{2}} \leq (m_1+m_2)^{2p-\frac{1+\gamma}{2}}\ 2^{-m_1-m_2}.$$

The cardinality of $\mathcal{W}^p(m_1, m_2)$ is bounded by $12 \cdot 2^{m_1+m_2}\, m_1^p\, m_2^p \cdot \max(m_1, m_2)^{2p} \leq 12 \cdot 2^{m_1+m_2}\, (m_1+m_2)^{4p}$. The number of possible wedges $W_m \in \mathcal{W}^p(m_1, m_2)$ derives from

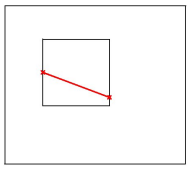
1. $\#\mathcal{R}^p(m_1, m_2) \leq 2^{m_1+m_2}\, m_1^p\, m_2^p$ choices for the corresponding rectangle $R(W)$,



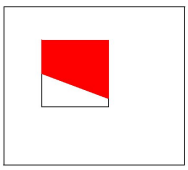
2. 6 choices for the 2 sides of the rectangle $R(W)$, which are cut by the dividing line of the wedge,

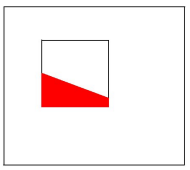


3. $\max(m_1, m_2)^{2p}$ choices for the possible positions of the intersections of the dividing line with $R(W)$ and



4. 2 choices between upper and lower, respectively between left and right wedge.





Hence with $\#\mathcal{W}^p(m_1, m_2) \leq 12 \cdot 2^{m_1+m_2}(m_1+m_2)^{4p}$, we obtain

$$\mathbb{P}\left(\sup_{W\in\mathcal{W}^p(m_1,m_2)} \frac{|B(W)|}{\sqrt{w}\sqrt{2\log\frac{1}{w}+\gamma\log\log\frac{1}{w}}} > 1\right) \leq C\,(m_1+m_2)^{6p-\frac{\gamma+1}{2}}$$

for some constant $C > 0$. For $W \in \cup_{m_1,m_2}\mathcal{W}^p(m_1, m_2)$, we get

$$\mathbb{P}\left(\sup_{m_1,m_2}\sup_{W\in\mathcal{W}^p(m_1,m_2)} \frac{|B(W)|}{\sqrt{w}\sqrt{2\log\frac{1}{w}+\gamma\log\log\frac{1}{w}}} > 1\right)$$

$$\leq C\sum_{m_1,\,m_2=1}^{\infty}(m_1+m_2)^{6p-\frac{\gamma+1}{2}}.$$

This sum converges if $6p - \frac{\gamma+1}{2} < -2$, i.e. if $\gamma > 12p + 3$. Thus for each $\varepsilon > 0$, there exists $m_0 = m_0(\varepsilon)$ so that for all $\gamma > 12p + 3$

$$\sum_{m_1+m_2\geq m_0}(m_1+m_2)^{6p-\frac{\gamma+1}{2}} < \varepsilon,$$

which means that for $\mathcal{W}^p(m_0) = \cup_{m_1+m_2\geq m_0}\mathcal{W}^p(m_1, m_2)$, we have

$$\mathbb{P}\left(\sup_{W\in\mathcal{W}^p(m_0)} \frac{|B(W)|}{\sqrt{w}\sqrt{2\log\frac{1}{w}+\gamma\log\log\frac{1}{w}}} > 1\right) < \varepsilon.$$

□

With the aid of this lemma, we can give the proof of the theorem.

Proof of Theorem 5.9, part 2
The idea of the proof is to approximate any wedge $W \in \mathcal{W}$ by a wedge $W_m \in \mathcal{W}^p(m_1, m_2)$ for specified $p, m_1, m_2 \in \mathbb{N}$. We will show that the error we make by this approximation is sufficiently small. The supremum over all $W \in \mathcal{W}^{\kappa-reg}$ with $|R(W)| \leq q_0$ is reduced to the supremum over all $W_m \in \mathcal{W}^p(m_0) \cap \mathcal{W}^{\kappa-reg}$, hence we can use Lemma 5.10.
Let $p \in \mathbb{N}$ be fixed. The choice of p is specified later. For an arbitrary wedge $W \in \mathcal{W}$ with corresponding rectangle $R(W) = [s_1, s_1 + h_1] \times [s_2, s_2 + h_2] \in \mathcal{R}$, we consider m_1 and m_2 with

$$(m_1+1)^p\, 2^{-(m_1+1)} < h_1 \leq m_1^p\, 2^{-m_1} \quad \text{and} \quad (m_2+1)^p\, 2^{-(m_2+1)} < h_2 \leq m_2^p\, 2^{-m_2}\,.$$

Let $W_m \in \mathcal{W}^p(m_1, m_2)$ be maximal with $W_m \subseteq W$, and for each $j \in \mathbb{N}$, let $W_{m+j} \in \mathcal{W}^p(m_1 + j, m_2 + j)$ be maximal with $W_{m+j} \subseteq W$. We will approximate W by W_m and estimate the error we make by this approximation.

First of all, we only consider those wedges $W \in \mathcal{W}$, for which the angle of the dividing line is either $\frac{\pi}{2}$ or $-\frac{\pi}{2}$. For such a wedge W, we have

$$W_{m+j} \subseteq W_{m+j+1},$$

$$W = W_m \cup \overline{\bigcup_{j=0}^{\infty} W_{m+j+1} \setminus W_{m+j}} \quad \text{and}$$

$$|B(W)| \leq |B(W_m)| + \sum_{j=0}^{\infty} |B(W_{m+j+1} \setminus W_{m+j})|.$$

For the sets $W_{m+j+1} \setminus W_{m+j}$ the random variable $B(W_{m+j+1} \setminus W_{m+j})$ is normally distributed with variance

$$|W_{m+j+1} \setminus W_{m+j}| \leq 2(h_1\, 2^{-m_2-j} + h_2)\, 2^{-m_1-j}$$

$$= \sigma^2(m_1, m_2, j) \leq 2^{-j+2}\, 2^{-m_1-m_2}\, (m_1 + m_2)^p.$$

Considering all possible wedges $W \in \mathcal{W}$, a problem may arise when we proceed from an approximation $W_{m+j} \in \mathcal{W}^p(m_1 + j, m_2 + j)$ to the next approximation $W_{m+j+1} \in \mathcal{W}^p(m_1 + j + 1, m_2 + j + 1)$. W_{m+j} may not be included in W_{m+j+1}, as illustrated for a triangle in Figure 5.3.

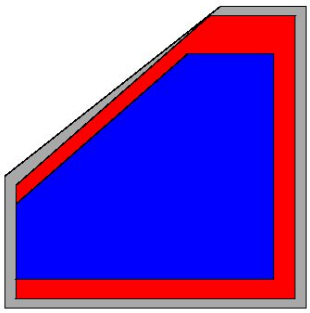


Figure 5.2: Original wedge $W \in \mathcal{W}$ and approximations W_m and W_{m+1}

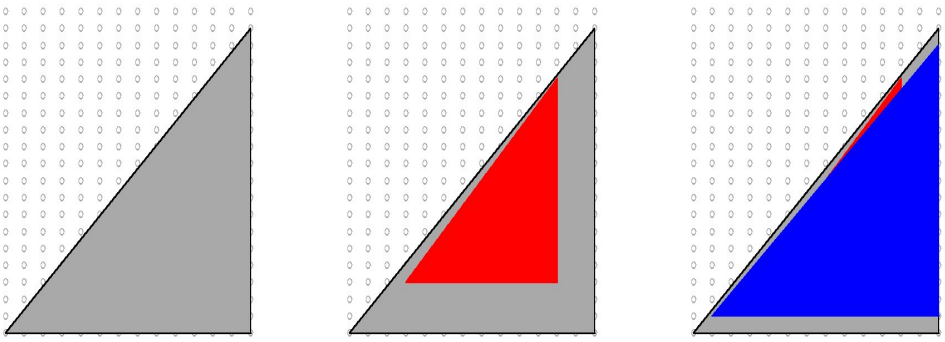


Figure 5.3: Original triangle W and approximations W_m and W_{m+1}

Therefore we consider the sets $\tilde{W}_m \subseteq \tilde{W}_{m+1} \subseteq \tilde{W}_{m+2} \subseteq \dots$ defined by

- $\tilde{W}_m = W_m$
- $\tilde{W}_{m+j+1} = \tilde{W}_{m+j} \cup W_{m+j+1}$.

The resulting sets $\tilde{W}_{m+j}$ need no longer be elements of $\mathcal{W}$. But we have

- $\tilde{W}_m \in \mathcal{W}^p(m_1, m_2)$ and $B(\tilde{W}_{m+j+1} \setminus \tilde{W}_{m+j}) \sim \mathcal{N}(0, |\tilde{W}_{m+j+1} \setminus \tilde{W}_{m+j}|)$ with

$$|\tilde{W}_{m+j+1} \setminus \tilde{W}_{m+j}| \le 2(h_1\, 2^{-m_2-j} + h_2)\, 2^{-m_1-j}$$
$$= \sigma^2(m_1, m_2, j) \le 2^{-j+2}\, 2^{-m_1-m_2}\, (m_1+m_2)^p,$$

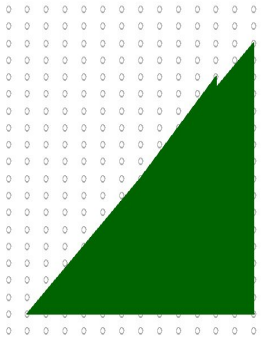


Figure 5.4: Set $\tilde{W}_{m+1}$ replacing the triangle W_{m+1}

- $\tilde{W}_{m+j+1} \subseteq \tilde{W}_{mj}$,
- $W = \tilde{W}_m \cup \overline{\bigcup_{j=0}^{\infty} \tilde{W}_{m+j+1} \setminus \tilde{W}_{m+j}}$ and
- $|B(W)| \leq |B(\tilde{W}_m)| + \sum_{j=0}^{\infty} |B(\tilde{W}_{m+j+1} \setminus \tilde{W}_{m+j})|$.

These are the same properties as for the wedges with angles $\frac{\pi}{2}$ and $-\frac{\pi}{2}$. We will only use these properties in the following, therefore from now on we write W_{m+j} also for $\tilde{W}_{m+j}$, having in mind that these sets are not necessarily elements of $\mathcal{W}$.

Using $B(W_{m+j+1} \setminus W_{m+j}) \sim \mathcal{N}(0, |W_{m+j+1} \setminus W_{m+j}|)$ with $|W_{m+j+1} \setminus W_{m+j}| \leq \sigma^2(m_1, m_2, j)$, we obtain for each $j \in \mathbb{N}_0$

$$\mathbb{P}\left(\frac{|B(W_{m+j+1} \setminus W_{m+j})|}{\sigma(m_1, m_2, j)} > \sqrt{2 \log \frac{1}{\sigma^2(m_1, m_2, j)} + \gamma' \log\log \frac{1}{\sigma^2(m_1, m_2, j)}} \right)$$

$$\leq 2^{-j+2} 2^{-m_1-m_2} \, (m_1 + m_2)^{p - \frac{1+\gamma'}{2}},$$

and hence for the supremum over all the sets $W_{m+j+1} \setminus W_{m+j}$ for $j = 0, 1, \ldots$

$$\mathbb{P}\left(\sup_j \frac{|B(W_{m+j+1} \setminus W_{m+j})|}{\sigma(m_1, m_2, j) \sqrt{2 \log \frac{1}{\sigma^2(m_1, m_2, j)} + \gamma' \log\log \frac{1}{\sigma^2(m_1, m_2, j)}}} > 1 \right)$$

$$\leq C \cdot 2^{-m_1-m_2} \, (m_1 + m_2)^{p - \frac{1+\gamma'}{2}} \quad \text{for some constant } C > 0.$$

We define the set $\mathcal{W}^{p\star}(m_1, m_2)$ containing all the wedges $W \in \mathcal{W}$ which yield the same pair (m_1, m_2) via $R(W)$. The number of resulting wedges W_m for $W \in \mathcal{W}^{p\star}(m_1, m_2)$ is given by $\#\mathcal{W}^p(m_1, m_2) \leq 12\,(m_1 + m_2)^{4p}\, 2^{m_1}\, 2^{m_2}$, and thus

$$\mathbb{P}\left(\sup_{W \in \mathcal{W}^{p\star}(m_1,m_2)} \sup_j \frac{|B(W_{m+j+1} \setminus W_{m+j})|}{\sigma(m_1, m_2, j)\sqrt{2\log\frac{1}{\sigma^2(m_1,m_2,j)} + \gamma' \log\log\frac{1}{\sigma^2(m_1,m_2,j)}}} > 1 \right)$$

$$\leq C'\,(m_1 + m_2)^{5p - \frac{1+\gamma'}{2}} \quad \text{for a constant } C' > 0.$$

Considering the supremum over all wedges $W \in \cup_{m_1,m_2} \mathcal{W}^{p\star}(m_1, m_2) = \mathcal{W}$, we get

$$\mathbb{P}\left(\sup_{W \in \cup_{m_1,m_2} W^{p\star}(m_1,m_2)} \sup_j \frac{|B(W_{m+j+1} \setminus W_{m+j})|}{\sigma(m_1, m_2, j)\sqrt{2\log\frac{1}{\sigma^2(m_1,m_2,j)} + \gamma' \log\log\frac{1}{\sigma^2(m_1,m_2,j)}}} > 1 \right)$$

$$\leq C' \sum_{m_1,\, m_2 = 1}^{\infty} (m_1 + m_2)^{5p - \frac{1+\gamma'}{2}}.$$

This sum converges for $5p - \frac{1+\gamma'}{2} < -2$, i.e. for $\gamma' > 3 + 10p$. Thus there exists $m_0 = m_0(\varepsilon)$, such that

$$\mathbb{P}\left(\sup_{W \in W^{p\star}(m_0)} \sup_j \frac{|B(W_{m+j+1} \setminus W_{m+j})|}{\sigma(m_1, m_2, j)\sqrt{2\log\frac{1}{\sigma^2(m_1,m_2,j)} + \gamma' \log\log\frac{1}{\sigma^2(m_1,m_2,j)}}} > 1 \right) \leq \varepsilon$$

with $W^{p\star}(m_0) := \cup_{m_1+m_2 \geq m_0} W^{p\star}(m_1, m_2)$.

The remaining computations are the same as the computations for rectangles carried out in Theorem 5.6. The only difference is that $|W| = \xi \cdot h_1 h_2$ for $0.5 \leq \xi \leq 1$ instead of $|R| = h_1 h_2$. This constant ξ does not play a role as we consider $h_1 h_2 \to 0$.

□

5.3 Embedding of Gaussian White Noise into Brownian Motion and Brownian Sheet

This section deals with the embedding of i.i.d. $\mathcal{N}(0,1)$-random variables, more precisely of scaled sums of these random variables, into the Brownian motion, respectively into the Brownian sheet. This embedding in combination with the modulus of continuity shown in Sections 5.1 and 5.2 justifies the multiresolution criterion for Gaussian white noise. We start with the one-dimensional case.

Theorem 5.11
Let $X_i \sim \mathcal{N}(0,1)$ be i.i.d. random variables for $1 \leq i \leq n$. The scaled partial sums

$$S_k = \frac{1}{\sqrt{n}} \sum_{i=1}^{k} X_i, \quad 1 \leq k \leq n$$

can be embedded in a Brownian motion $B = (B_t)_{t\in[0,1]}$ with $B(0) = 0$.

Proof
For $1 \leq k \leq n$, we have

$$B\left(\frac{k}{n}\right) = \sum_{i=1}^{k} \left(B\left(\frac{i}{n}\right) - B\left(\frac{i-1}{n}\right)\right).$$

We define the random variables Y_i and Z_i for $1 \leq i \leq n$ by

$$Y_i := B\left(\frac{i}{n}\right) - B\left(\frac{i-1}{n}\right) \qquad \text{and} \qquad Z_i := \sqrt{n}\, Y_i.$$

Then $Y_i \sim \mathcal{N}(0, \frac{1}{n})$ and $Z_i \sim \mathcal{N}(0,1)$ are i.i.d. random variables. Hence the random variables $B\left(\frac{k}{n}\right)$ can also be written as scaled partial sums of the i.i.d. $\mathcal{N}(0,1)$-random variables Z_i:

$$B\left(\frac{k}{n}\right) = \frac{1}{\sqrt{n}} \sum_{i=1}^{k} Z_i =: \tilde{S}_k.$$

Thus $(S_1, \ldots, S_n)$ and $(\tilde{S}_1, \ldots, \tilde{S}_n)$ have the same distribution.

□

Using this embedding, we obtain for an interval $I = [i,j] \subseteq [1,n]$ with $|I| = j-i+1$

$$\begin{aligned}
\frac{1}{\sqrt{j-i+1}} \sum_{l=i}^{j} X_l &= \frac{\sqrt{n}}{\sqrt{j-i+1}} \frac{1}{\sqrt{n}} \sum_{l=i}^{j} X_l = \frac{\sqrt{n}}{\sqrt{j-i+1}} (S_j - S_i) \\
&\overset{D}{=} \frac{\sqrt{n}}{\sqrt{j-i+1}} (\tilde{S}_j - \tilde{S}_i) = \frac{\sqrt{n}}{\sqrt{j-i+1}} \left(B\left(\frac{j}{n}\right) - B\left(\frac{i-1}{n}\right)\right) \\
&= \frac{B\left(\frac{j}{n}\right) - B\left(\frac{i-1}{n}\right)}{\sqrt{\frac{j-i+1}{n}}} = \frac{B(t) - B(s)}{\sqrt{t-s}}
\end{aligned}$$

with $0 \leq \frac{i-1}{n} =: s < t := \frac{j}{n} \leq 1$. Hence part 3 of Theorem 5.3 yields for all $\varepsilon > 0$

$$\max_{[i,j]\in\tilde{\mathcal{I}}} \frac{\left|\sum_{l=i}^{j} X_l\right|}{\sqrt{|I|}\sqrt{\delta \log \frac{n}{|I|}}} \leq \sup_{0\leq s<t\leq 1,\, t-s\neq 1} \frac{|B(t)-B(s)|}{\sqrt{t-s}\sqrt{\log \frac{1}{t-s}}} \leq 1$$

with a probability greater than $1-\varepsilon$ for some constant $\delta = \delta(\varepsilon) > 2$, where $\tilde{\mathcal{I}} \subset \mathfrak{P}(\{1,2,\ldots,n\})$ consists of all the subintervals of $[1,n]$, except for the interval $I = [1,n]$, which corresponds to the interval $[s,t]$ with $t-s=1$.
Thus for an i.i.d. sequence Z_i, $1 \leq i \leq n$ with $Z_i \sim \mathcal{N}(0,\sigma^2)$, we obtain with a probability greater than $1-\varepsilon$

$$\max_{I\in\tilde{\mathcal{I}}} \frac{\left|\sum_{i\in I} Z_i\right|}{\sqrt{|I|}\sqrt{\delta \log \frac{n}{|I|}}} \leq \sigma \quad \text{for some constant } \delta > 2.$$

With $|I| \geq 1$, we obtain the simpler version

$$\max_{I\in\mathcal{I}} \frac{\left|\sum_{i\in I} Z_i\right|}{\sqrt{|I|}} \leq \sigma\sqrt{\delta \log n}$$

with the general threshold $\sigma\sqrt{\delta \log n}$ for each interval $I \in \mathcal{I} = \tilde{\mathcal{I}} \cup [1,n]$.

Part 4 of Theorem 5.3 yields for all $\varepsilon > 0$ the following inequalities:

$$\max_{I\in\tilde{\mathcal{I}}} \frac{\left|\sum_{i\in I} Z_i\right|}{\sqrt{|I|}\sqrt{2\log \frac{n}{|I|} + \gamma \log\log \frac{n}{|I|}}} \leq \sigma \quad \text{and}$$

$$\max_{I\in\mathcal{I}} \frac{\left|\sum_{i\in I} Z_i\right|}{\sqrt{|I|}} \leq \sigma\sqrt{2\log n + \gamma \log\log n}$$

for some constant $\gamma = \gamma(\varepsilon) > 5$ with a probability greater than $1-\varepsilon$.

Hence for all $\varepsilon > 0$ there exist some constants $\delta > 2$ and $\gamma > 5$, so that the following inequalities hold for Gaussian white noise with a probability greater than $1-\varepsilon$.

1. $\max_{I\in\tilde{\mathcal{I}}} \frac{\left|\sum_{i\in I} Z_i\right|}{\sqrt{|I|}\sqrt{\delta \log \frac{n}{|I|}}} \leq \sigma$,
2. $\max_{I\in\mathcal{I}} \frac{\left|\sum_{i\in I} Z_i\right|}{\sqrt{|I|}} \leq \sigma\sqrt{\delta \log n}$,
3. $\max_{I\in\tilde{\mathcal{I}}} \frac{\left|\sum_{i\in I} Z_i\right|}{\sqrt{|I|}\sqrt{2\log \frac{n}{|I|} + \gamma \log\log \frac{n}{|I|}}} \leq \sigma$,

4. $\max_{I\in\mathcal{I}} \frac{\left|\sum_{i\in I} Z_i\right|}{\sqrt{|I|}} \leq \sigma\sqrt{2\log n + \gamma \log\log n}$.

If we want to include the interval $I = [1, n]$ also in the first and third inequality, we need to insert constants, e.g. $e > 1$ and $e^e > e$, and obtain

1′. $\max_{I\in\mathcal{I}} \frac{\left|\sum_{i\in I} Z_i\right|}{\sqrt{|I|}\sqrt{\delta \log \frac{en}{|I|}}} \leq \sigma$,

3′. $\max_{I\in\mathcal{I}} \frac{\left|\sum_{i\in I} Z_i\right|}{\sqrt{|I|}\sqrt{2\log \frac{n}{|I|} + \gamma \log\log \frac{e^e n}{|I|}}} \leq \sigma$.

The constants $\delta > 2$ and $\gamma > 5$ can be determined by simulations.
In Section 2.6, the second inequality is used to check whether all the residuals $r_n(x_i) = y_i - \hat{f}_n(x_i)$ of an estimator $\hat{f}_n$ are i.i.d. $\mathcal{N}(0, \sigma^2)$. The other inequalities yield variations of the multiresolution criterion. The versions using $\frac{n}{|I|}$ result in a lower threshold for large intervals intervals. This might be desirable in some applications, but mostly the general threshold is good enough. Therefore we have restricted ourselves to the simplest version, using the general threshold $\sigma\sqrt{\delta \log n}$.

The two-dimensional embedding is stated without proof. The connection between the scaled sums $\frac{1}{\sqrt{N}} \sum_{k=1}^{i} \sum_{l=1}^{j} X_{ij} = B_{m,n}([0, \frac{i}{m}] \times [0, \frac{j}{n}]) \stackrel{D}{=} B\left(\frac{i}{m}, \frac{j}{n}\right)$ is already mentioned as a remark to Definition 5.8.

Theorem 5.12
Let $X_{ij} \sim \mathcal{N}(0, 1)$ be i.i.d. random variables for $1 \leq i \leq m$ and $1 \leq j \leq n$ and $N = m \cdot n$. The scaled partial sums

$$S_{ij} = \frac{1}{\sqrt{N}} \sum_{k=1}^{i} \sum_{l=1}^{j} X_{ij}, \quad 1 \leq i \leq m,\, 1 \leq j \leq n$$

can be embedded in a Brownian sheet $B = (B(s))_{s\in[0,1]^2}$.

As for one dimension, this can be combined with the modulus of continuity for rectangles and for wedges. Hence for all $\varepsilon > 0$ we obtain for i.i.d. random variables $Z_{ij} \sim \mathcal{N}(0, \sigma^2)$ with $1 \leq i \leq m$, $1 \leq j \leq n$ and $N = m \cdot n$ that the following inequalities hold with a probability greater than $1 - \varepsilon$.

1. $\max_{P\in\tilde{\mathcal{P}}} \frac{\left|\sum_{(i,j)\in P} Z_{ij}\right|}{\sqrt{|P|}\sqrt{\delta \log \frac{N}{|P|}}} \leq \sigma$,

2. $\max_{P\in\mathcal{P}} \frac{\left|\sum_{(i,j)\in P} Z_{ij}\right|}{\sqrt{|P|}} \leq \sigma\sqrt{\delta \log N}$,

3. $\max_{P\in\tilde{\mathcal{P}}} \frac{\left|\sum_{(i,j)\in P} Z_{ij}\right|}{\sqrt{|P|}\sqrt{2\log\frac{N}{|P|}+\gamma\log\log\frac{N}{|P|}}} \leq \sigma,$

4. $\max_{P\in\mathcal{P}} \frac{\left|\sum_{(i,j)\in P} Z_{ij}\right|}{\sqrt{|P|}} \leq \sigma\sqrt{2\log N+\gamma\log\log N},$

where $\mathcal{P}$ is either the partition into dyadic squares $\mathcal{P}_S$ or the wedge partition $\mathcal{P}_W$ and $\tilde{\mathcal{P}} = \mathcal{P} \setminus [1,m]\times[1,n]$.

If we want to include the whole image $P = [1,m]\times[1,n]$ also in the first and third inequality, we need to insert constants, e.g. $e > 1$ and $e^e > e$, and obtain

1′. $\max_{P\in\mathcal{P}} \frac{\left|\sum_{(i,j)\in P} Z_{ij}\right|}{\sqrt{|P|}\sqrt{\delta\log\frac{eN}{|P|}}} \leq \sigma,$

3′. $\max_{P\in\mathcal{P}} \frac{\left|\sum_{(i,j)\in P} Z_{ij}\right|}{\sqrt{|P|}\sqrt{2\log\frac{N}{|P|}+\gamma\log\log\frac{e^eN}{|P|}}} \leq \sigma.$

In Section 3.3.2, the second inequality is used for the two-dimensional multiresolution criterion. The other inequalities yield variations.

Chapter 6

Future Work

This last chapter contains some ideas about future research in the field of nonparametric regression by inhomogeneous diffusion.

6.1 Generalization of the Model

First of all the underlying model could be generalized. We recall the model assumptions we have made:

$$Y_i = f(x_i) + \sigma Z_i, \qquad Z_i \sim \mathcal{N}(0,1).$$

Two possible generalizations can be easily combined with the diffusion estimator: heteroschedastic noise and other kinds of noise. In the following these ideas are sketched in the one-dimensional setting.

6.1.1 Heteroschedastic Noise

In the case of heteroschedastic noise, the standard deviation σ is not constant, but varies from place to place. This results in the model

$$Y_i = f(x_i) + \sigma_i Z_i, \qquad Z_i \sim \mathcal{N}(0,1).$$

For independent random variables $V_i \sim \mathcal{N}(0, \sigma_i^2)$, the random variables

$$S_I^2 = \sum_{i \in I} \frac{V_i^2}{\sigma_i^2} \quad \text{for } I \in \mathcal{I}$$

are supposed to be χ^2-distributed with $|I|$ degrees of freedom. This property can be used to determine an estimate $\hat{\sigma}$ for the standard deviation of the heteroschedastic noise $V_i = \sigma_i Z_i$.

For data y_i generated under the model $Y_i = f(x_i) + \sigma_i Z_i$ with a smooth function f, the differences $y_i - y_{i-1}$ are close to $\sigma_i Z_i - \sigma_{i-1} Z_{i-1}$. Thus we have approximately

$$\frac{y_i - y_{i-1}}{\sqrt{2}} \sim \mathcal{N}(0, \sigma_i^2),$$

and the heteroschedastic noise level σ can be estimated from the values $v_i := \frac{y_i - y_{i-1}}{\sqrt{2}}$. $\hat{\sigma}$ is accepted as an estimate for the noise level, if

$$qu_{\chi^2_{|I|}}(0.005) \leq \sum_{i \in I} \frac{v_i^2}{\hat{\sigma}(x_i)^2} \leq qu_{\chi^2_{|I|}}(0.995) \qquad \forall I \in \mathcal{I},$$

where $qu_{\chi^2_{|I|}}(\gamma)$ is the γ-quantile for the χ^2-distribution with $|I|$ degrees of freedom.

In the next step, $\hat{\sigma}_i = \hat{\sigma}(x_i)$ is inserted in the model to obtain $Y_i = f(x_i) + \hat{\sigma}_i Z_i$ as a new model. The heteroschedastic estimator $\hat{f}_a^{\,het}$ is computed in the same way as the inhomogeneous diffusion estimator $\hat{f}_a$, but with the heteroschedastic multiresolution conditions

$$|w_I^{het}| \leq \sqrt{\delta \log n} \qquad \forall I \in \mathcal{I}$$

with $w_I^{het} := \frac{1}{\sqrt{|I|}} \sum_{i \in I} \frac{y_i - \hat{f}_n(x_i)}{\hat{\sigma}_i}$.

6.1.2 Other Kinds of Noise

Instead of assuming that the noise is Gaussian, we could also assume other kinds of noise. Dümbgen and Kovac [DK05b] have modified the multiresolution criterion for Cauchy noise, Poisson noise and Binary noise. The resultant models are

$$Y_i = f(x_i) + \sigma Z_i, \qquad Z_i \sim \mathcal{C}(0,1),$$

$$Y_i \sim Po(\lambda_i) \text{ with } \lambda_i = f\left(\frac{x_i - a}{b}\right),$$

$$Y_i \sim Bin(1, p_i) \text{ with } p_i = f(x_i) - a,$$

where $a = \min_{t \in [0,1]} f(t)$ and $b = \max_{t \in [0,1]} f(t)$.

6.2 Higher Dimensions

Another idea is to increase the dimension of the problem. Because of the fast numerical computation of the diffusion estimator, it is possible to transfer the procedure to three-dimensional images and perhaps even to higher dimensions. In

the following, we give a short outline of the three-dimensional problem.
The three-dimensional model is given by

$$Y_{ijk} = f(\mathbf{x}_{ijk}) + \sigma\, Z_{ijk}, \quad \mathbf{x}_{ijk} \in [0,1]^3, \quad Z_{ijk} \text{ i.i.d. } \mathcal{N}(0,1).$$

For a three-dimensional image with $N := (m+1)\cdot(n+1)\cdot(o+1)$ voxels, the points $\mathbf{x}_{ijk}$ can also be interpreted as a triple of three natural numbers $\mathbf{x}_{ijk} = (m_i, n_j, o_k)$ with $0 \le m_i \le m$, $0 \le n_j \le n$, $0 \le o_k \le o$.
The three-dimensional inhomogeneous diffusion process is given by

$$\frac{\partial}{\partial t} u(\mathbf{x}, t) = a(\mathbf{x})\ \Delta u(\mathbf{x}, t)$$

with $\mathbf{x} = (x_1, x_2, x_3) \in \Omega = [0, m] \times [0, n] \times [0, o]$, $t \ge 0$ and $a(\mathbf{x}) \ge 0$. Here $\Delta = \frac{\partial^2}{\partial x_1^2} + \frac{\partial^2}{\partial x_2^2} + \frac{\partial^2}{\partial x_3^2}$ is the three-dimensional Laplace operator.

$\hat{f}_a$, the three-dimensional diffusion estimator for the data $(\mathbf{x}_{ijk}, y_{ijk})$, is defined as the solution $u(\mathbf{x}, 1)$ of the three-dimensional inhomogeneous diffusion process stopped at $\tau = 1$ with

- diffusivity $a(\mathbf{x}) \ge 0$,
- starting values $u(\mathbf{x}_{ijk}, 0) = y_{ijk}$,
- homogeneous Neumann conditions $\nabla u(\mathbf{x}, t) = 0$ for $\mathbf{x} \in \partial\Omega$.

The diffusivity $a(\mathbf{x})$ is the local smoothing parameter of the estimator $\hat{f}_a$. It is determined by a three-dimensional version of the multiresolution criterion. For the three-dimensional multiresolution coefficients

$$w_C = \frac{1}{\sqrt{|C|}} \sum_{\mathbf{x}_{ijk} \in C} r(\mathbf{x}_{ijk}) \qquad \text{with } r(\mathbf{x}_{ijk}) = y_{ijk} - \hat{f}_a(\mathbf{x}_{ijk})$$

we have to check whether

$$|w_C| \le \sigma_N \sqrt{\delta \log N}.$$

Theoretically this inequality has to be checked for all connected subsets $C \subseteq \Omega$, but in practice we only go through all C within a three-dimensional partition of Ω.
We suggest the partition $\mathcal{C}$ into dyadic cubes. For an image of $2^n \times 2^n \times 2^n$ voxels, $\mathcal{C}$ consists of the elements

$$C = \left[i_1 \cdot 2^{j-1},\, (i_1+1)\cdot 2^j - 1\right] \times \left[i_2 \cdot 2^{j-1},\, (i_2+1)\cdot 2^j - 1\right] \times \left[i_3 \cdot 2^{j-1},\, (i_3+1)\cdot 2^j - 1\right],$$

for $j = 1, \ldots n$ and $i_1, i_2, i_3 = 0, \ldots, 2^{n-j}$.

The value of the constant $\delta > 0$ is derived from the modulus of continuity of the three-dimensional Brownian sheet, which has to be investigated.

Appendix A

Datasets

In Appendix A, we have collected all the datasets used in this thesis. Both for the one-dimensional and the two-dimensional case, there are two types of datasets, namely the artificial datasets and the real datasets.
The artificial datasets have been generated under the model

$$Y_i = f(x_i) + \sigma Z_i,$$

where $Z_i \sim \mathcal{N}(0,1)$ are i.i.d. random variables and σ is the standard deviation of the noise.
The real datasets consist of data (x_i, y_i), which have been measured in different applications. For these datasets we have to estimate the noiselevel and look for a smooth representation of the data.

A.1 One-dimensional Datasets

A.1.1 The Sine and the Sinepeak Datasets

The Sine Dataset is an artificial dataset. It is generated by

$$Y_i = \sin(2\pi x_i) + 0.1 \cdot Z_i,$$

where $Z_i \sim \mathcal{N}(0,1)$ i.i.d. and $x_i = i \cdot 10^{-5}$ for $0 \leq i \leq 10^5$.

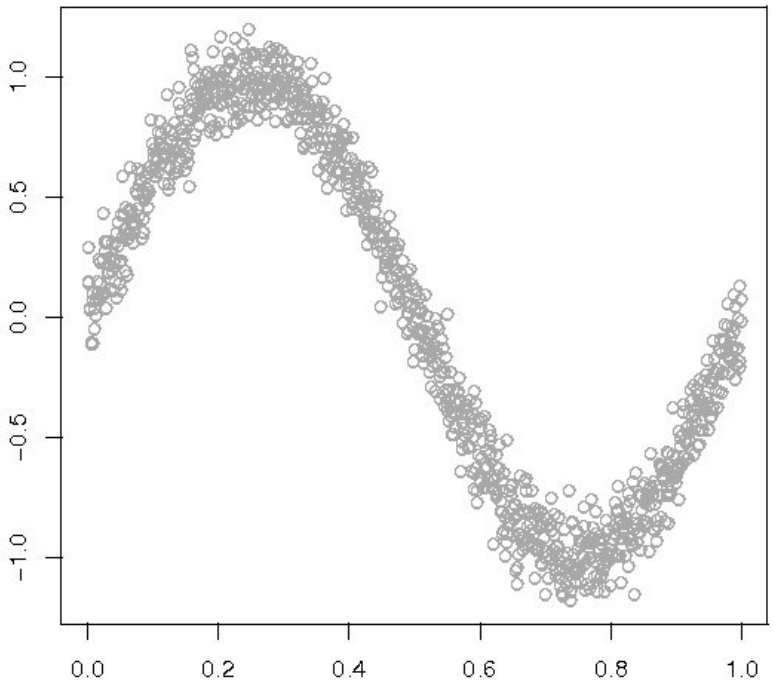


Figure A.1: Sine dataset

In order to obtain a dataset with local differences in smoothness, a peak is added to the sine. The Sinepeak dataset is generated by

$$Y_i = \sin(2\pi x_i) + \exp(-(40 \cdot (x_i - 0.5))^2) + 0.1 \cdot Z_i,$$

where $Z_i \sim \mathcal{N}(0,1)$ i.i.d. and $x_i = i \cdot 10^{-5}$ for $0 \leq i \leq 10^5$.

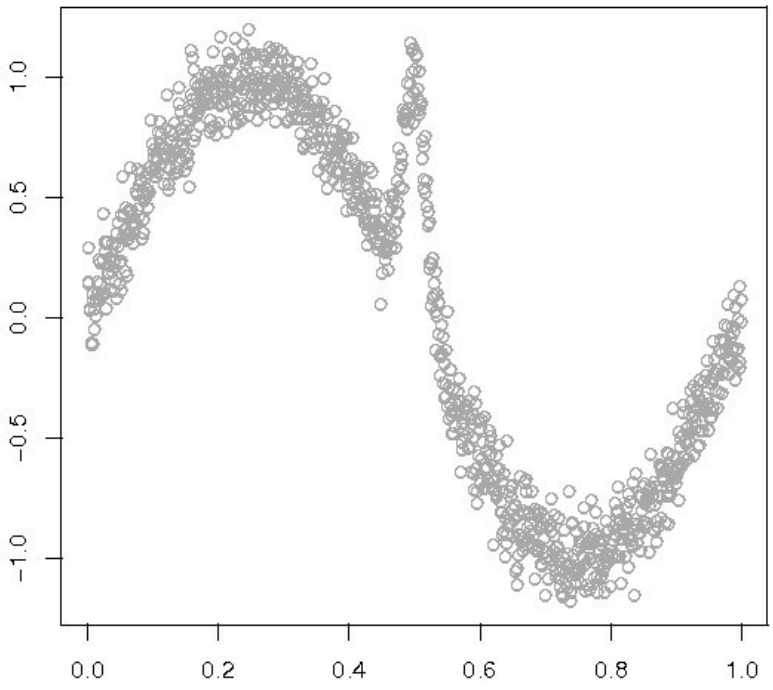


Figure A.2: Sinepeak dataset

Based on these datasets of size 10^5, we obtain datasets of the sizes $n = 10^4$, $n = 10^3$ and $n = 100$, consisting of every 10th, 100th and 1000th data point. In Chapters 1 and 2, the Sine and the Sinepeak datasets are used for the different sample sizes. Here we have plotted the ones of sample size $n = 10^3$.

A.1.2 The Jump Dataset

The Jump dataset is a very simple dataset containing a discontinuity. It is generated by

$$Y_i = \mathbb{1}_{[0.5,1]}(x_i) + 0.1 \cdot Z_i,$$

where $Z_i \sim \mathcal{N}(0,1)$ i.i.d. and $x_i = \frac{i}{99}$ for $0 \leq i \leq 99$.

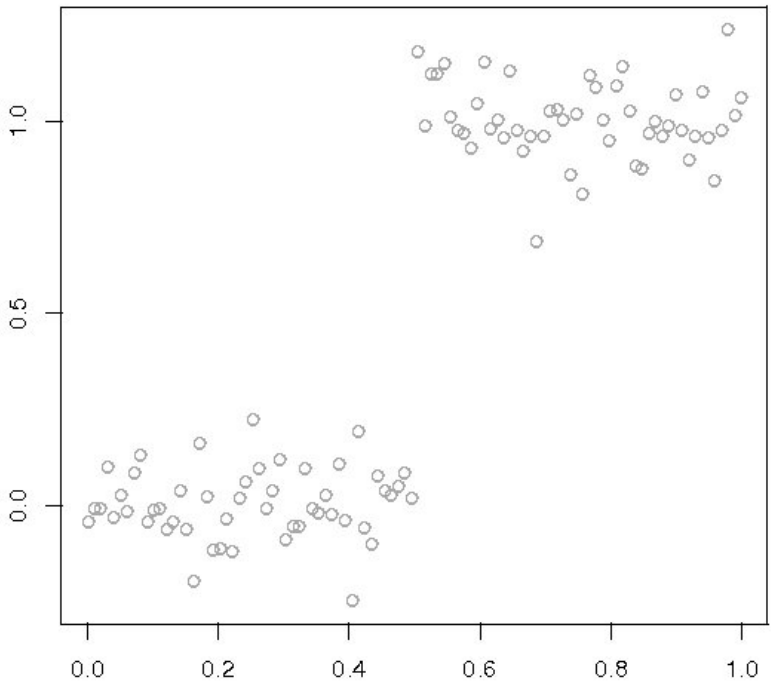


Figure A.3: Jump dataset

A.1.3 The X-ray Diffractogram Dataset

The X-ray Diffractogram dataset is kindly made available to use by Dieter Mergel from the Physics Department at the University of Duisburg-Essen. It consists of 7001 data points, which come from an experiment where an X-ray has been sent on a material and the number of photons, diffracted in different angles between 15° and 85° has been recorded. The spectrum, in particular the location, power and width of the peaks, is then used to obtain information about the material. For a thorough analysis of this dataset we refer to [DMMM07].

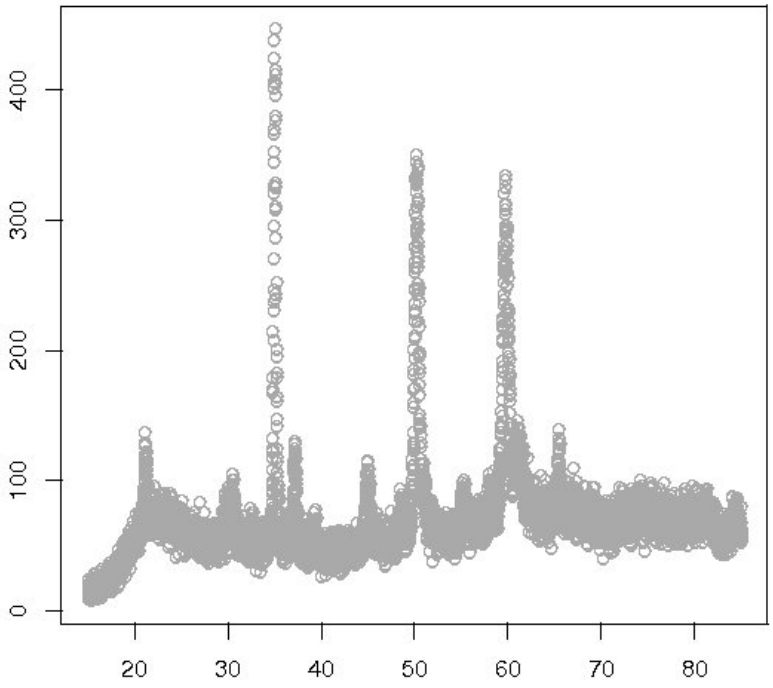


Figure A.4: X-ray Diffractogram dataset

A.1.4 The Bumps, Heavisine, Doppler and Blocks Datasets

The four test signals of Donoho and Johnstone, namely the Bumps, Heavisine, Doppler and Blocks test signal, are often used to test the performance of an estimator. They contain various difficulties, such as varying smoothness and discontinuities. Here we have generated four datasets, each of sample size 2048. The design points are $x_i = \frac{i}{2047}$, $0 \leq i \leq 2047$.
The functions are scaled so that the standard deviation is 1, and the noise level is $\sigma = 0.2$. This means that the root signal-to-noise ratio RSNR, which is the ratio of the standard deviation of the function values to the standard deviation of the noise, is 5.

The Bumps Dataset

The Bumps dataset is generated by

$$Y_i = f_{Bumps}(x_i) + 0.2 \cdot Z_i,$$

where $Z_i \sim \mathcal{N}(0,1)$ i.i.d. and

$$f_{Bumps}(x) = 1.503227 \cdot \sum_{j=1}^{11} \frac{h_j}{(1 + |(x - x_j)/w_j|)^4} \quad \text{with}$$

$\mathbf{x} = (\,0.1,\ 0.13,\ 0.15,\ 0.23,\ 0.25,\ 0.40,\ 0.44,\ 0.65,\ 0.76,\ 0.78,\ 0.81\,)$,
$\mathbf{h} = (\,4,\ 5,\ 3,\ 4,\ 5,\ 4.2,\ 2.1,\ 4.3,\ 3.1,\ 5.1,\ 4.2\,)$,
$\mathbf{w} = (\,0.005,\ 0.005,\ 0.006,\ 0.01,\ 0.01,\ 0.03,\ 0.01,\ 0.01,\ 0.005,\ 0.008,\ 0.005\,)$.

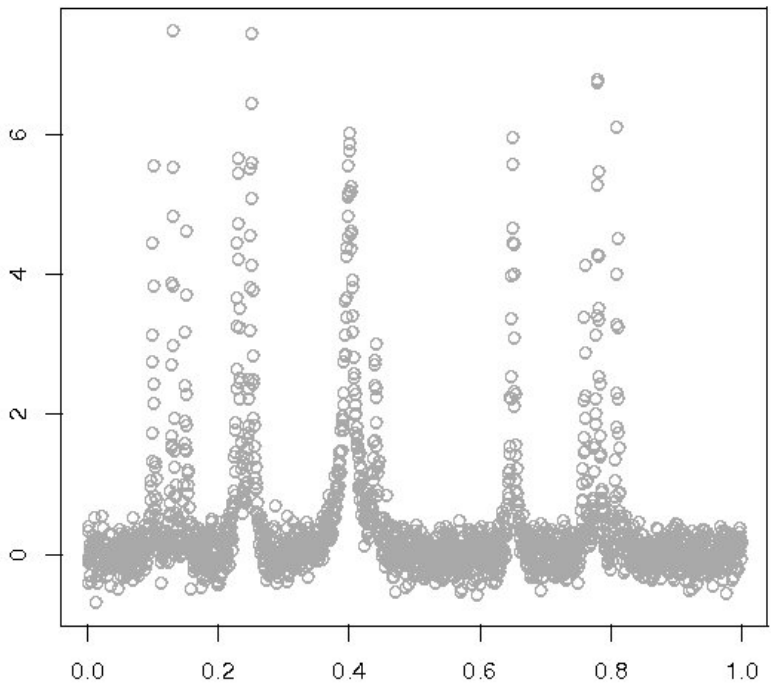


Figure A.5: Bumps dataset

The Heavisine Dataset

The Heavisine dataset is generated by

$$Y_i = f_{Heavisine}(x_i) + 0.2 \cdot Z_i,$$

where $Z_i \sim \mathcal{N}(0,1)$ i.i.d. and

$$f_{Heavisine}(x) = 0.3367195 \cdot (4\sin(4\pi x) - \text{sgn}(x - 0.3) - \text{sgn}(0.72 - x)).$$

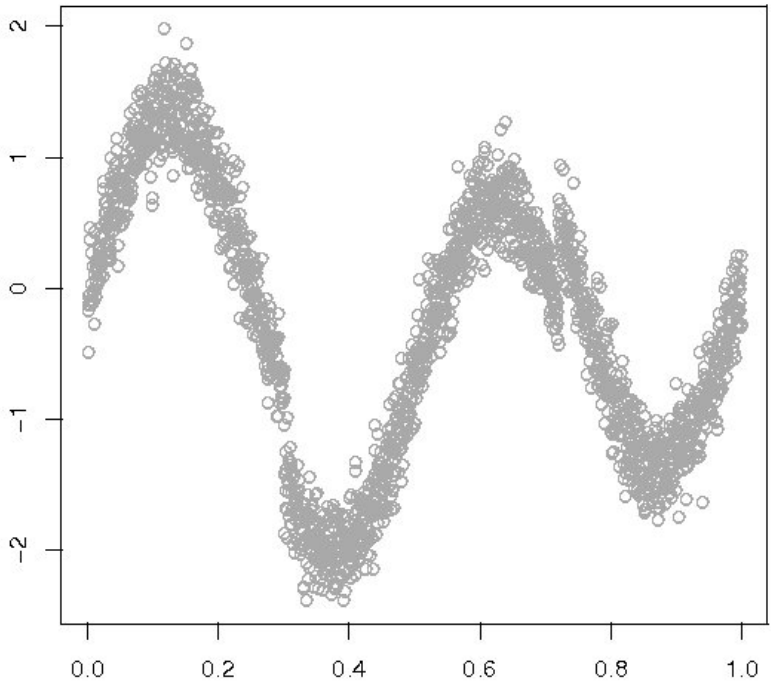


Figure A.6: Heavisine dataset

The Doppler Dataset

The Doppler dataset is generated by

$$Y_i = f_{Doppler}(x_i) + 0.2 \cdot Z_i,$$

where $Z_i \sim \mathcal{N}(0,1)$ i.i.d. and

$$f_{Doppler}(x) = 3.460228 \cdot \sqrt{x(1-x)} \sin\left(\frac{2.1\pi}{x+0.05}\right).$$

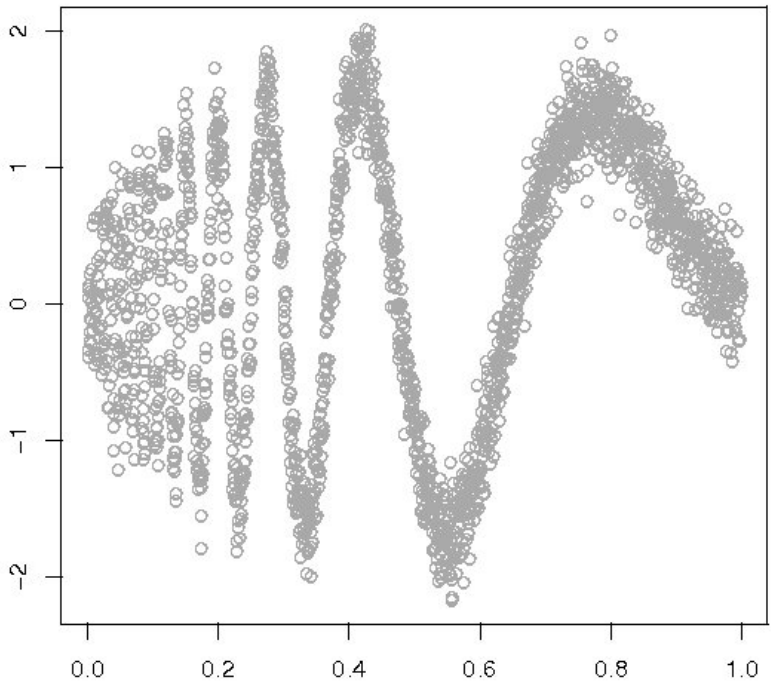


Figure A.7: Doppler dataset

The Blocks Dataset

The Blocks dataset is generated by

$$Y_i = f_{Blocks}(x_i) + 0.2 \cdot Z_i,$$

where $Z_i \sim \mathcal{N}(0,1)$ i.i.d. and

$$f_{Blocks}(x) = 0.5046483 \cdot \sum_{j=1}^{11} h_j \, \frac{1 + \mathrm{sgn}(x - x_j)}{2} \quad \text{with}$$

$\mathbf{x} = (\,0.1,\ 0.13,\ 0.15,\ 0.23,\ 0.25,\ 0.40,\ 0.44,\ 0.65,\ 0.76,\ 0.78,\ 0.81\,)$,
$\mathbf{h} = (\,4,\ -5,\ 3,\ -4,\ 5,\ -4.2,\ 2.1,\ 4.3,\ -3.1,\ 5.1,\ -4.2\,)$.

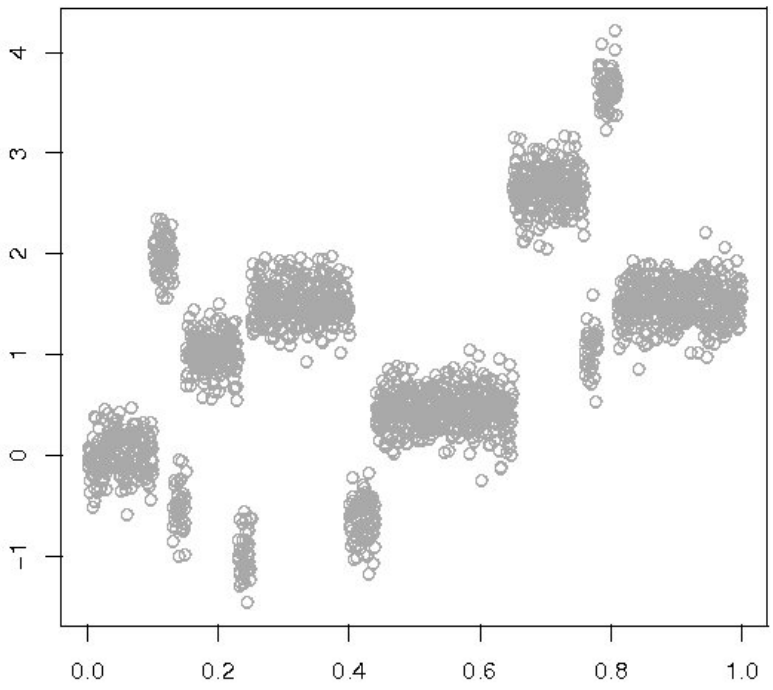


Figure A.8: Blocks datasets

A.2 Two-dimensional Datasets

A.2.1 The Mexican Hat Dataset

The Mexican Hat function

$$f_{Mexican}(\mathbf{x}) = \frac{1}{\sqrt{2\pi}\rho^3}\left(1 - ||\mathbf{x}||^2\right) \exp\left(-\frac{||\mathbf{x}||^2}{2\rho^2}\right)$$

is a smooth two-dimensional function. The Mexican Hat dataset consists of 51×51 pixels and is generated by

$$Y_{ij} = f_{Mexican}(\mathbf{x_{ij}}) + 0.01\, Z_{ij}$$

with $\rho = 0.9$, $Z_{ij} \sim \mathcal{N}(0,1)$ i.i.d. and equidistant design points in $[-5,5] \times [-5,5]$.

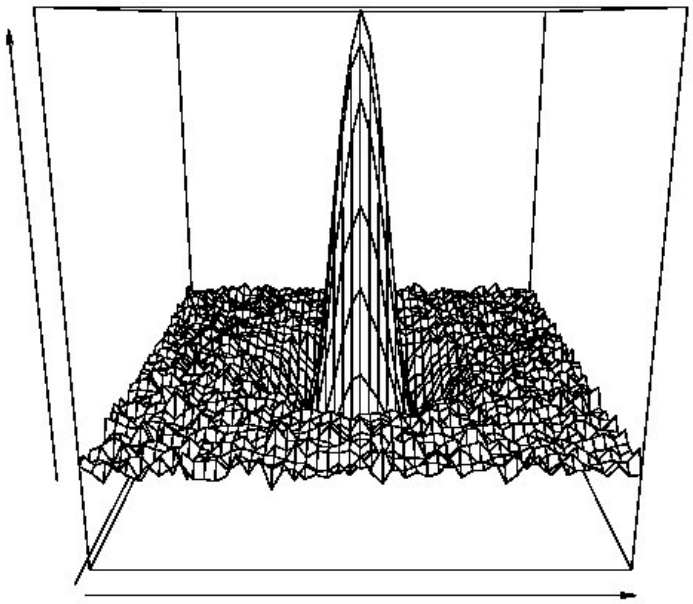


Figure A.9: Mexican Hat dataset

A.2.2 The Peak Dataset

This dataset imitates a single peak which might be contained in a two-dimensional spectroscopy dataset. Here the difference in smoothness is rather extreme. The Peak dataset contains 256×256 pixels and is generated by

$$Y_{ij} = f_{Peak}(\mathbf{x}_{ij}) + 40\, Z_{ij}$$

with $Z_{ij} \sim \mathcal{N}(0,1)$ i.i.d., $\mathbf{x}_{ij} = (i,j)$, $1 \leq i,j \leq 256$ and

$$f_{Peak}(i,j) = 1000 \cdot \exp\left(-5000\left(\frac{(i-150)^2}{256^2} + \frac{(j-100)^2}{256^2}\right)\right).$$

In Figure A.10 and in Chapter 3 only every fourth pixel of the dataset is plotted to obtain a better visibility.

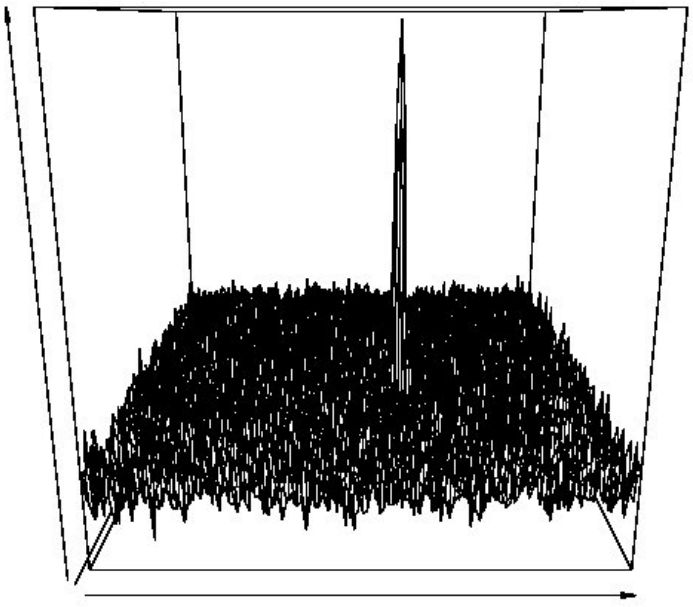


Figure A.10: Peak dataset

A.2.3 The Quantum Computing Dataset

This dataset is kindly made available to us by Emre Togan from the Department of Applied Physics at Harvard University. The 200×200 pixel image displays the photoluminescence in a diamond sample which is observed through a confocal microscope.
At each pixel, the number of photons, which are detected from that point in space during $10\,ms$, is counted and re-scaled to obtain the number of photons per second. The research group at Harvard University is interested in the location of the bright spots, which indicate nitrogene-vacancy centres. The NV centres are further investigated since they are good candidates for qubits.

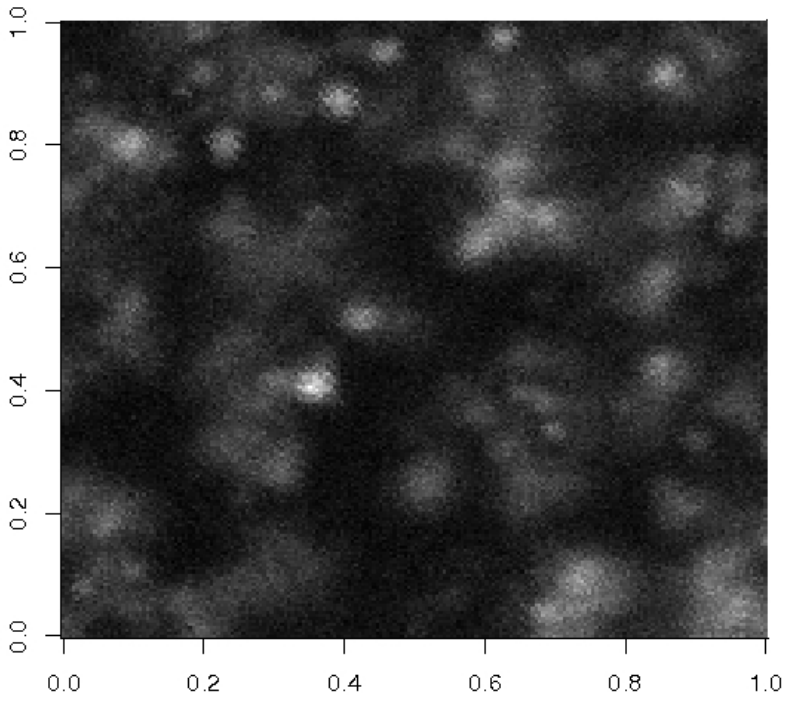


Figure A.11: Quantum Computing dataset

Appendix B

Source Code

Appendix B contains the source code for the diffusion estimator for the one-dimensional and the two-dimensional setting. Here we restrict ourselves to the simplest versions of $\hat{f}_a$ with automatically selected diffusivity a.
All the other version of the diffusion estimator can be found on the webpage http://www.stat-math.uni-essen.de/∼stichtenoth.

B.1 Source Code for the One-dimensional Diffusion Estimator $\hat{f}_a$

In the one-dimensional setting we have chosen the version of $\hat{f}_a$ where the selection of the local diffusivity takes into account only the dyadic intervals.
On the webpage, one additionally finds the source code for the estimator $\hat{f}_\tau$ with the global smoothing parameter τ, for the diffusion estimator $\hat{f}_a$ with local, but fixed diffusivity and for the version of $\hat{f}_a$ with automatically selected local diffusivity where all intervals are considered, as well as the corresponding versions for the variation $\hat{f}^\star$ of the diffusion estimator.

Listing B.1: R-code for the one-dimensional diffusion estimator $\hat{f}_a$

```
#diffest1d computes the diffusion estimator with
#automatically selected local diffusivity
#using the multiresolution criterion on dyadic intervals
#INPUT:
#y=noisy data
#delta=2.3 by default (>2)
#afactor=25/(max(y)-min(y)) by default,
#yields initial value a=afactor*sigman
```

```
#OUTPUT:
#est=smooth estimator, diff=diffusivity

diffest1d <- function(y,delta=2.3,afactor=25/(max(y)-min(y)))
  {
    x<-seq(0,1,le=length(y))
    n <- length(y)
    fn <- double(n)
    a <- double(n)
    p <- rep(1,le=n)
    #estimate of the noise level
    sigman <- median(abs(diff(y)))*1.481 / sqrt(2)
    print(c("estimated noiselevel",sigman))
    a <- rep(afactor*sigman,le=n)
    #general threshold for the multiresolution criterion
    thresh <- sigman*sqrt(delta*log(n))

    #computation of the diffusion estimator
    dyn.load("diffest1d.so")
    tmp <- .C("diffest1d",
              as.integer(n),
              as.double(x),
              as.double(y),
              as.double(fn),
              as.integer(p),
              as.double(a),
              as.double(thresh))
    list(diff=tmp[[6]], est=tmp[[4]])
  }
```

Listing B.2: C-code for the one-dimensional diffusion estimator $\hat{f}_a$

```
#include <stdio.h>
#include <math.h>
#include <R.h>

void diffest1d (int *n, double *x, double *y, double *fn,
                int *p, double *a, double *thresh)
{
  int count, i, j;
```

```
9    int *stp;
10   double A[*n*3];
11   double C[*n];
12   void settridiag();
13   void solvetridiag();
14   void checkMRdyadic_1d();
15   void adapt1_1d();
16
17   stp=malloc(sizeof(int));
18
19   count=0;
20   do
21     {
22       count+=1;
23       settridiag(y,a,n,&A,&C);
24       solvetridiag(A,C,n,fn);
25       checkMRdyadic_1d (y,fn,p,n,thresh);
26       adapt1_1d(p,a,n,stp);
27     }while(*stp==1);
28
29   printf("Iteration steps of the diffusion estimator: %i\n",
30          count);
31   free(stp);
32 }
33
34
35 /*********************************************************
36  **** settridiag sets the values for the discretized ****
37  ****     version  Ax+C=0 of the diffusion process    ****
38  ****          d/dt u(x,t)= a(x) d2/dx2 u(x,t)        ****
39  *********************************************************/
40
41 void settridiag(double *y, double *a, int *n, double *A,
42                 double *C)
43 {
44   int i;
45
46   A[0]=0.0;
47   A[*n]=1+2*a[0]*pow(*n-1,2);
48   A[2*(*n)]=-2*a[0]*pow(*n-1,2);
```

```
49
50    for (i=1; i<(*n-1); i++)
51      {
52        A[i]=-a[i]*pow(*n-1,2);;
53        A[*n+i]=1+2*a[i]*pow(*n-1,2);
54        A[2*(*n)+i]=A[i];
55      }
56
57    A[*n-1]=-2*a[*n-1]*pow(*n-1,2);
58    A[*n+*n-1]=1+2*a[*n-1]*pow(*n-1,2);
59    A[2*(*n)+*n-1]=0.0;
60
61    for (i=0;i<*n;i++)
62      C[i]=-y[i];
63  }
64
65
66 /*************************************************************
67  *** solvetridiag solves the tridiagonal system Ax+C=0 **
68  *************************************************************/
69
70 void solvetridiag(double *A, double *C, int *n, double *fn)
71 {
72    double r[*n];
73    double s[*n-1], t[*n-1], f[*n];
74    double u1, v, w, z;
75    int i;
76
77    for (i=0; i<*n-1; i++)
78      {
79        if(fabs(A[*n+i])/(fabs(A[*n+i])+fabs(A[2*(*n)+i])) >=
80            fabs(A[1+i])/
81            (fabs(A[1+i])+fabs(A[*n+1+i])+fabs(A[2*(*n)+1+i])))
82          {
83            r[i]=A[*n+i];
84            s[i]=A[2*(*n)+i];
85            t[i]=0.0;
86            f[i]=C[i];
87            u1=A[1+i];
88            v=A[*n+1+i];
```

```
89              w=A[2*(*n)+1+i];
90              z=C[i+1];
91            }
92          else
93            {
94              r[i]=A[1+i];
95              s[i]=A[*n+1+i];
96              t[i]=A[2*(*n)+1+i];
97              f[i]=C[i+1];
98              u1=A[*n+i];
99              v=A[2*(*n)+i];
100             w=0.0;
101             z=C[i];
102           }
103         A[i]=u1/r[i];
104         A[*n+1+i]=v-A[i]*s[i];
105         A[2*(*n)+1+i]=w-A[i]*t[i];
106         C[i+1]=z-A[i]*f[i];
107       }
108
109     r[*n-1]=A[*n+*n-1];
110     f[*n-1]=C[*n-1];
111     fn[*n-1]=-f[*n-1]/r[*n-1];
112     fn[*n-2]=-(f[*n-2]+s[*n-2]*fn[*n-1])/r[*n-2];
113
114     for(i=*n-3;i>=0;i--)
115       {
116         fn[i]=-(f[i]+s[i]*fn[i+1]+t[i]*fn[i+2])/r[i];
117       }
118 }
119
120
121 /*********************************************************
122  ****       checkMRdyadic_1d checks MR condition       ****
123  ****              on dyadic intervals I               ****
124  ****     if not satisfied in I, p[i]=1 for i in I     ****
125  *********************************************************/
126
127 void checkMRdyadic_1d (double *y, double *fn, int *p,
128                        int *n, double *thresh)
```

```
129 {
130   int i, interlength, i1, i2, j;
131   double R[*n+1];
132
133   R[0]=0.;
134   for(i=0;i<*n;i++)
135     {
136       p[i]=0;
137       R[i+1]=R[i]+y[i]-fn[i];
138     }
139
140   for (interlength=1; interlength<=*n; interlength*=2)
141     {
142       for (i1=0,i2=interlength;i1<*n;i1=i2,i2+=interlength)
143         {
144           if(i2>*n)
145             i2=*n;
146           if((fabs(R[i2]-R[i1])/sqrt(i2-i1)) > *thresh)
147             {
148               for (j=i1;j<i2;j++)
149                 p[j]=1;
150             }
151         }
152     }
153 }
154
155
156 /*************************************************
157  ****  adapt1_1d adapts the smoothing parameter *par ****
158  ****  it is lowered (*0.8) at places i with p[i]=1  ****
    ********
159  *************************************************/
160
161 void adapt1_1d(int *p,double *par,int *n, int *stp)
162 {
163   int i;
164   *stp=0;
165   for (i=0;i<*n;i++)
166     if(p[i]==1)
167       {
```

```
168             par[i]=0.8*par[i];
169             *stp=1;
170           }
171 }
```

B.2 Source Code for the Two-dimensional Diffusion Estimator $\hat{f}_a$

Our choice in the two-dimensional setting has been the version of $\hat{f}_a$, where the local diffusivity is adapted on the partition $\mathcal{P}_S$ of dyadic squares.
On the webpage, one additionally finds the source code for the estimator $\hat{f}_\tau$ with the global smoothing parameter τ, for the diffusion estimator $\hat{f}_a$ with local, but fixed diffusivity and for the versions of $\hat{f}_a$ where the local diffusivity is adapted on two levels or on the wedge partition $\mathcal{P}_W$. All these versions are also available for the variation $\hat{f}^\star$ of the diffusion estimator.

Listing B.3: R-code for the two-dimensional diffusion estimator $\hat{f}_a$

```
1  #diffest2d computes the diffusion estimator with
2  #automatically selected local diffusivity
3  #using the multiresolution criterion on dyadic squares
4  #INPUT:
5  #Y=noisy image
6  #delta=2.3 by default (>2)
7  #afactor=500/(max(Y)-min(Y)) by default,
8  #yields initial value A=afactor*sigman
9  #SOR=1.9 by default, successive overrelaxation parameter
10 #values of SOR in (0,2), SOR=1:normal Gauss-Seidel algorithm
11 #GSiterations=number of iteration steps of the SOR-algorithm
12 #to solve the PDE (default=100)
13 #MRiterations=maximal number of iteration steps for the
14 #selection of the diffusivity (default=100)
15 #OUTPUT:
16 #U=processed image,
17 #A=diffusivity
18
19 diffest2d<-function(Y, delta=2.3,SOR=1.9, GSiterations=100,
20                     afactor=500/(max(Y)-min(Y)),
```

```
                    MRiterations=100)
  {
    dimx<-dim(Y)[2]
    dimy<-dim(Y)[1]
    #estimate of the noise level
    M<-matrix(0,dim(Y)[1]-1,dim(Y)[2]-1)
    for(i in 1:(dim(Y)[1]-1))
      for(j in 1:(dim(Y)[2]-1))
        M[i,j]<-abs(Y[i+1,j+1]-Y[i+1,j]-Y[i,j+1]+Y[i,j])
    m<-matrix2vector(M)
    sigman<-1.481/2.*median(m)
    print(c("Estimated noiselevel:",sigman))
    #general threshold for the multiresolution criterion
    thresh<-sigman*(sqrt(delta*log((dimy-1)*(dimx-1))));
    y<-matrix2vector(Y)
    a<-rep(afactor*sigman,le=length(Y))
    u<-double(length(Y))

    dyn.load("diffest2d.so")
    tmp<-.C("diffest2d",
            as.double(y),
            as.double(a),
            as.double(u),
            as.integer(dimx),
            as.integer(dimy),
            as.integer(GSiterations),
            as.double(SOR),
            as.double(thresh),
            as.integer(MRiterations))

    U<-vector2matrix(tmp[[3]],dim(Y)[1],dim(Y)[2])
    A<-vector2matrix(tmp[[2]],dim(Y)[1],dim(Y)[2])
    list(U=U,A=A)
  }


#transforms matrix A into vector v of length dim(A)
matrix2vector<-function(A)
  {
    v<-double(length(A))
```

```
61      for(i in 0:(dim(A)[2]-1))
62        for (j in 1:(dim(A)[1]))
63          v[i*(dim(A)[1])+j]=A[j,i+1]
64      v
65    }
66
67
68  #transforms vector v into matrix of dimensions dimy*dimx
69  vector2matrix<-function(v,dimy,dimx)
70  {
71   if(dimx*dimy != length(v))
72    print("This vector cannot be transformed into a matrix
73  of these dimensions!")
74   A<-matrix(0,dimy,dimx)
75   i<-integer
76   j<-integer
77   for (i in 0:(dim(A)[2]-1))
78     for (j in 1:(dim(A)[1]))
79       A[j,i+1]=v[i*(dim(A)[1])+j]
80   A
81   }
```

Listing B.4: C-code for the two-dimensional diffusion estimator $\hat{f}_a$

```
1  #define MAX(x,y) (((x)<(y))?(y):(x))
2  #define MIN(x,y) (((x)<(y))?(x):(y))
3  #define M_PI 3.14159265358979323846
4  #include <math.h>
5  #include <stdio.h>
6  #include <R.h>
7
8
9  /*********************************************************
10  **** the structure node stores the elements of the  ****
11  ****          partition into dyadic squares          ****
12  *********************************************************/
13
14 struct node
15 {
16   int x,y,w,h,level,stpnode;
```

```
17    struct node *ll, *lr, *ul, *ur;
18  };
19
20
21  void diffest2d(double *y, double *a, double *u, int *dimx,
22                 int *dimy, int *GSiterations,double *w,
23                 double *thresh, int *MRiterations)
24  {
25    int i,count;
26    int *p,*stop;
27    double *residuals;
28
29    void heat2dfix();
30    void adapt1_2d();
31    void checkMRsquares_2d();
32    struct node *createtree();
33    struct node *root;
34
35    stop=malloc(sizeof(int));
36    residuals=malloc((*dimx*(*dimy))*sizeof(double));
37    p=malloc((*dimx*(*dimy))*sizeof(int));
38    root=malloc(sizeof(struct node));
39
40    // creates the tree of dyadic rectangles
41    root = (struct node *)createtree(1,1,*dimx,*dimy,0);
42
43    count=0;
44    do
45      {
46        count+=1;
47        heat2dfix(y,a,u,dimx,dimy,GSiterations,w);
48        for (i=0;i<(*dimx*(*dimy));i++)
49          residuals[i]=u[i]-y[i];
50        *stop=0;
51        checkMRsquares_2d(residuals,dimx,dimy,thresh,p,
52                          root,stop);
53        if(*stop==0)
54          {
55            adapt1_2d(p,a,dimx,dimy);
56            if(count==*MRiterations)
```

```
57              {
58                 printf("Abort after %i th iteration step.\n",
59                        *MRiterations);
60                 *stop=1;
61              }
62          }
63      }while(*stop==0);
64    free(root);
65    printf("Iteration steps of the diffusion estimator: %i\n",
66           count);
67 }
68
69
70 /*******************************************************
71  **** createtree builds up recursively the structure ****
72  ****       node containing all dyadic squares       ****
73  *******************************************************/
74
75 struct node *createtree(int x, int y, int w, int h,
76                         int level)
77 {
78    int left, right, upper, lower;
79    struct node *n;
80
81    n=(struct node *) malloc(sizeof(struct node));
82
83    n->w = w;
84    n->h=h;
85    n->x=x;
86    n->y=y;
87    n->level=level;
88    n->stpnode=0;
89
90    if ((w>1)||(h>1))
91      {
92        left=w/2;
93        right=w-left;
94        lower=h/2;
95        upper=h-lower;
96        if ((left>0) && (lower>0))
```

```
97          {
98            n->ll =(struct node *)
99              createtree(x,y,left,lower,level+1);
100         }
101       if ((right>0) && (lower>0))
102         {
103           n->lr =(struct node *)
104             createtree(x+left,y,right,lower,level+1);
105         }
106       if ((left>0) && (upper>0))
107         {
108           n->ul =(struct node *)
109             createtree(x,y+lower,left,upper,level+1);
110         }
111       if ((right>0) && (upper>0))
112         {
113           n->ur =(struct node *)
114             createtree(x+left,y+lower,right,upper,level+1);
115         }
116     }
117   return n;
118 }
119
120
121 /*********************************************************
122  ****checkMRsquares_2d checks MR condition on dyadic ****
123  ****          squares via sum and checksquare          ****
124  *********************************************************/
125
126 void checkMRsquares_2d(double *residuals, int *dimx,
127                        int *dimy, double *thresh, int *p,
128                        struct node *root, int *stop)
129 {
130   void checksquare();
131   double *ressum;
132   void sum();
133   int i;
134
135   ressum=malloc((*dimx*(*dimy))*sizeof(double));
136
```

```
137   sum(residuals,ressum,dimx,dimy);
138
139   *stop=1;
140   for (i=0;i<(*dimx*(*dimy));i++)
141     p[i]=0;
142
143   // checks MR condition recursively on all dyadic squares
144   checksquare (root,ressum,thresh,p,dimx,dimy,stop);
145
146   free(ressum);
147 }
148
149
150 /*********************************************************
151  ****        sum calculates the matrix of sums          ****
152  ****     Summ[i,j]= sum_{k=0;l=0}^{k=i;l=j} M[i,l]     ****
153  *********************************************************/
154
155 void sum (double *m, double *summ, int *dimx, int *dimy)
156 {
157   int i,j;
158   double colsum[*dimy*(*dimx)];
159
160   // sums the columns: Colsum[i,j]=sum_{l=0}^{l=j} M[i,l]
161   for (i=0;i<(*dimx);i++)
162     {
163       colsum[i*(*dimy)]=m[i*(*dimy)];
164       for (j=1;j<(*dimy);j++)
165         colsum[i*(*dimy)+j]=colsum[i*(*dimy)+j-1]
166           +m[i*(*dimy)+j];
167     }
168
169   // sums everything left and up from (i,j):
170   // Summ[i,j]= sum_{k=0;l=0}^{k=i;l=j} M[i,l]
171   for (j=0;j<(*dimy);j++)
172     summ[j]=colsum[j];
173   for (i=1;i<(*dimx);i++)
174     for (j=0;j<(*dimy);j++)
175       summ[i*(*dimy)+j]=summ[(i-1)*(*dimy)+j]
176         +colsum[i*(*dimy)+j];
```

```
177 }
178
179
180 /*********************************************************
181  **** checksquare checks MR condition on a square n  ****
182  ****       and recursively on its subsquares        ****
183  *********************************************************/
184
185 void checksquare (struct node *n, double *ressum,
186                   double *thresh, int *p, int *dimx,
187                   int *dimy, int *stop)
188 {
189   int i,j;
190   double one,two,three,four;
191
192   n->stpnode=0;
193   one=ressum[((n->x+n->w-1)-1)*(*dimy)+(n->y+n->h-1)-1];
194
195   if(n->x==1)
196     {
197       three=0.;
198       four=0.;
199       if(n->y==1)
200         two=0.;
201       else
202         two=ressum[((n->x+n->w-1)-1)*(*dimy)+(n->y-1)-1];
203     }
204   else if(n->y==1)
205     {
206       two=0.;
207       four=0.;
208       three=ressum[((n->x-1)-1)*(*dimy)+(n->y+n->h-1)-1];
209     }
210   else
211     {
212       two=ressum[((n->x+n->w-1)-1)*(*dimy)+(n->y-1)-1];
213       three=ressum[((n->x-1)-1)*(*dimy)+(n->y+n->h-1)-1];
214       four= ressum[((n->x-1)-1)*(*dimy)+(n->y-1)-1];
215     }
216
```

```
217    if(fabs(one-two-three+four)/(sqrt((n->w)*(n->h)))>*thresh)
218      {
219        *stop=0;
220        for (i=n->x;i<=n->x+n->w-1;i++)
221          for (j=n->y;j<=n->y+n->h-1;j++)
222            p[(i-1)*(*dimy)+j-1]=1;
223        n->stpnode=1;
224      }
225
226    if(((n->w)>1)&&((n->stpnode)==0))
227      {
228        checksquare(n->ul,ressum,thresh,p,dimx,dimy,stop);
229        checksquare(n->ur,ressum,thresh,p,dimx,dimy,stop);
230        if (n->h>1)
231          {
232            checksquare(n->ll,ressum,thresh,p,dimx,dimy,stop);
233            checksquare(n->lr,ressum,thresh,p,dimx,dimy,stop);
234          }
235      }
236    else if(n->h>1 &&n->stpnode==0)
237      {
238        checksquare(n->lr,ressum,thresh,p,dimx,dimy,stop);
239        checksquare(n->ur,ressum,thresh,p,dimx,dimy,stop);
240      }
241  }
242
243
244  /*********************************************************
245   **** adapt1_2d adapts the local smoothing parameter ****
246   ****  it is lowered (*0.8) at places i with p[i]=1  ****
247   *********************************************************/
248
249  void adapt1_2d(int *p,double *par,int *dimx,int *dimy)
250  {
251    int i;
252    for (i=0;i<(*dimx*(*dimy));i++)
253      if(p[i]==1)
254        par[i]=0.8*par[i];
255  }
256
```

```
257
258 /*************************************************************
259  ****      heat2dfix computes the solution of the          ****
260  ****         inhomogeneous diffusion process              ****
261  ****        d/dt u(x,t) = a(x) d^2/dx^2 u(x,t)            ****
262  ****         with fixed local diffusivity a(x)            ****
263  *************************************************************/
264
265 void heat2dfix(double *y, double *a, double *u, int *dimx,
266                int *dimy, int *GSiterations, double *w)
267 {
268   int i,j,ite;
269
270     for(i=0;i<(*dimx);i++)
271         for(j=0;j<(*dimy);j++)
272           u[i*(*dimy)+j]=0.;
273
274     for(ite=1;ite<*GSiterations;ite++)
275       {
276         for(j=1;j<(*dimy-1);j++)
277           u[j]=
278             (1-*w)*u[j]+*w*(y[j]+(u[j-1]+2*u[(*dimy)+j]
279                                   +u[j+1])*a[j])/(4*a[j]+1);
280         u[0]=
281           (1-*w)*u[0]+*w*(y[0]+(2*u[*dimy]
282                                 +2*u[1])*a[0])/(4*a[0]+1);
283         u[*dimy-1]=
284           (1-*w)*u[*dimy-1]
285           +*w*(y[*dimy-1]+(2*u[*dimy-2]+2*u[*dimy+*dimy-1])
286                 *a[*dimy-1])/(4*a[*dimy-1]+1);
287         for(j=1;j<(*dimy-1);j++)
288           u[(*dimx-1)*(*dimy)+j]=
289             (1-*w)*u[(*dimx-1)*(*dimy)+j]
290             +*w*(y[(*dimx-1)*(*dimy)+j]
291                   +(u[(*dimx-1)*(*dimy)+(j-1)]
292                     +u[(*dimx-1)*(*dimy)+(j+1)]
293                     +2*u[(*dimx-2)*(*dimy)+j])
294                   *a[(*dimx-1)*(*dimy)+j])
295               /(4*a[(*dimx-1)*(*dimy)+j]+1);
296         u[(*dimx-1)*(*dimy)]=
```

```
297             (1-*w)*u[(*dimx-1)*(*dimy)]
298             +*w*(y[(*dimx-1)*(*dimy)]+(2*u[(*dimx-1)*(*dimy)+1]
299                                       +2*u[(*dimx-2)*(*dimy)])
300                     *a[(*dimx-1)*(*dimy)])
301             /(4*a[(*dimx-1)*(*dimy)]+1);
302         u[(*dimx-1)*(*dimy)+(*dimy-1)]=
303             (1-*w)*u[(*dimx-1)*(*dimy)+(*dimy-1)]
304             +*w*(y[(*dimx-1)*(*dimy)+(*dimy-1)]
305                     +(2*u[(*dimx-1)*(*dimy)+(*dimy-2)]
306                       +2*u[(*dimx-2)*(*dimy)+(*dimy-1)])
307                     *a[(*dimx-1)*(*dimy)+(*dimy-1)])
308             /(4*a[(*dimx-1)*(*dimy)+(*dimy-1)]+1);
309         for(i=1;i<(*dimx-1);i++)
310             {
311                 for(j=1;j<(*dimy-1);j++)
312                     u[i*(*dimy)+j]=
313                         (1-*w)*u[i*(*dimy)+j]
314                         +*w*(y[i*(*dimy)+j]+(u[i*(*dimy)+(j-1)]
315                                             +u[(i+1)*(*dimy)+j]
316                                             +u[i*(*dimy)+(j+1)]
317                                             +u[(i-1)*(*dimy)+j])
318                                 *a[i*(*dimy)+j])/(4*a[i*(*dimy)+j]+1);
319                 u[i*(*dimy)]=
320                     (1-*w)*u[i*(*dimy)]
321                     +*w*(  y[i*(*dimy)]+(u[(i+1)*(*dimy)]
322                                         +2*u[i*(*dimy)+(1)]
323                                         +u[(i-1)*(*dimy)])
324                             *a[i*(*dimy)])/(4*a[i*(*dimy)]+1);
325                 u[i*(*dimy)+(*dimy-1)]=
326                     (1-*w)*u[i*(*dimy)+(*dimy-1)]
327                     +*w*(y[i*(*dimy)+(*dimy-1)]
328                             +(2*u[i*(*dimy)+(*dimy-2)]
329                               +u[(i+1)*(*dimy)+(*dimy-1)]
330                               +u[(i-1)*(*dimy)+(*dimy-1)])
331                             *a[i*(*dimy)+(*dimy-1)])
332                     /(4*a[i*(*dimy)+(*dimy-1)]+1);
333             }
334         }
335 }
```

List of Figures

List of Tables

Bibliography

[Ale86] K.S. Alexander. Sample Moduli for Set-indexed Gaussian Processes. *Annals of Probability*, 14(2):598–611, 1986.

[BEGS94] M. Brockmann, J. Engel, T. Gasser, and B. Seifert. Fast Algorithms for Nonparametric Curve Estimation. *Journal of Computational and Graphical Statistics*, 3(2):192–213, 1994.

[Bil79] P. Billingsley. *Probability and Measure*. John Wiley, 1979.

[Bre65] J.E. Bresenham. Algorithm for Computer Control of a Digital Plotter. *IBM Systems Journal*, 4(1):25–30, 1965.

[CDT+06] L. Childress, M.V. Gurudrev Dutt, J.M. Taylor, A.S. Zibrov, F. Jelezko, J. Wachtrup, P.R. Hemmer, and M.D. Lukin. Coherent Dynamics of Coupled Electron and Nuclear Spin Qubits in Diamond. *Science*, 314:281–285, 2006.

[CES59] K.L. Chung, P. Erdös, and T. Sirao. On the Lipschitz´s Condition for Brownian Motions. *Journal of the Mathematical Society of Japan*, 11:263–274, 1959.

[DCJ+07] M.V. Gurudev Dutt, L. Childress, L. Jiang, E. Togan, J. Maze, F. Jelezko, A.S. Zibrov, P.R. Hemmer, and M.D. Lukin. Quantum Register Based on Individual Electronic and Nuclear Spin Qubits in Diamond. *Science*, 316:1312–1316, 2007.

[DJ94] D.L. Donoho and I.M. Johnstone. Ideal Spatial Adaptation via Wavelet Shrinkage. *Biometrika*, 81:425–455, 1994.

[DJKP95] D.L. Donoho, I.M. Johnstone, G. Kerkyacharian, and D. Picard. Wavelet Shrinkage: Asymptotia? with Discussion. *Journal of the Royal Statistical Society, Series B*, 57:301–369, 1995.

[DK01] P.L. Davies and A. Kovac. Local Extremes, Runs, Strings and Multiresolution. *Annals of Statistics*, 29:1–65, 2001.

[DK04] P.L. Davies and A. Kovac. Densities, Spectral Densities and Modality. *Annals of Statistics*, 32:1093–1136, 2004.

[DK05a] P.L. Davies and A. Kovac. *ftnonpar: Features and Strings for Nonparametric Regression*, 2005. R package version 0.1-4.

[DK05b] L. Dümbgen and A. Kovac. Extensions of Smoothing via Taut Strings. Technical report, 2005.

[DKM08] P.L. Davies, A. Kovac, and M. Meise. Nonparametric Regression, Confidence Regions and Regularization. *Annals of Statistics*, 2008. To appear. arXiv: 0711.0690 [math.ST].

[DMMM07] P.L. Davies, M. Meise, D. Mergel, and T. Mildenberger. Residual Based Localisation and Quantification of Peaks in X-ray Diffractograms. Technical report, SFB 475, Department of Statistics, University of Dortmund, 2007.

[Don95] D.L. Donoho. Denoising via Soft Thresholding. *IEEE Transactions on Information Theory*, 41:613–627, 1995.

[DS01] L. Dümbgen and V.G. Spokoiny. Multiscale Testing of Qualitative Hypotheses. *Annals of Statistics*, 29(1):124–152, 2001.

[EMKA05] R.J. Epstein, F.M. Mendoza, Y.K. Kato, and D.D. Awschalom. Anisotropic Interactions of a Single Spin and Dark-spin Spectroscopy in Diamond. *Nature Physics*, 1:94–98, 2005.

[FDFW07] F. Friedrich, L. Demaret, H. Führ, and K. Wicker. Efficient Moment Computation over Polygonal Domains with an Application ro Rapid Wedgelet Approximation. *SIAM Journal of Scientific Computing*, 29(2):842–863, 2007.

[Fri05] F. Friedrich. *Complexity Penalized Segmentations in 2D*. PhD thesis, Technical University of Munich, 2005.

[GS94] P.J. Green and B.W. Silverman. *Nonparametric Regression and Generalized Linear Models*. Monographs on Statistics and Applied Probability 58. Chapman and Hall, 1994.

[Här90] W. Härdle. *Applied Nonparametric Regression*. Cambridge University Press, 1990.

[Her97] E. Herrmann. Local Bandwidth Choice in Kernel Regression Estimation. *Journal of Graphical and Computational Statistics*, 6:3554, 1997.

[HM85] W. Härdle and J.S. Marron. Optimal Bandwidth Selection in Nonparametric Regression Function Estimation. *Annals of Statistics*, 13:1465–1481, 1985.

[HM03] E. Herrmann and M. Mächler. *lokern: Kernel Regression Smoothing with Local or Global Plug-in Bandwidth*, 2003. R package version 1.0-4.

[KA00] P. Knabner and L. Angermann. *Numerik partieller Differentialgleichungen*. Springer, 2000.

[Ken05] T. Kennedy. Dark Spins Come to Light. *Nature Physics*, 1:79–80, 2005.

[Kov02] A. Kovac. Robust Nonparametric Regression and Modality. *Development in Statistics*, pages 218–227, 2002.

[Lan03] H.P. Langtangen. *Computational Partial Differential Equations*. Springer, 2003.

[Lév54] P. Lévy. *Théorie de l´Addition des Variables Aléatoires*. Gauthier-Villars, 2ème edition, 1954.

[Mei04] M. Meise. *Residual Based Selection of Smoothing Parameters*. PhD thesis, University of Duisburg-Essen, 2004.

[Nad64] E.A. Nadaraya. On Estimating Regression. *Theory of Probability and its Applications*, 10:186–190, 1964.

[Nas93] G.P. Nason. The Wavethresh Package; Wavelet Transform and Thresholding Software for S. Available from the StatLib Archive, 1993.

[NH95] A.N. Netravali and B.G. Haskell. *Digital Pictures: Representation, Compression, and Standards*. Plenum Press, 2nd edition, 1995.

[NKM06] G.P. Nason, A. Kovac, and M. Mächler. *wavethresh: Software to Perform Wavelet Statistics and Transforms*, 2006. R package version 2.2-9.

[NS94] G.P. Nason and B.W. Silverman. The Discrete Wavelet Transform in S. *Journal of Computational and Graphical Statistics*, 3(2):163–191, 1994.

[OP73] S. Orey and W.E. Pruitt. Sample Functions of the N-parameter Wiener Process. *Annals of Probability*, 1(1):138–163, 1973.

[PM87] P. Perona and J. Malik. Scale Space and Edge Detection Using Anisotropic Diffusion. *Proceedings of the IEEE Computer Society Workshop on Computer Vision, Miami Beach*, pages 16–22, 1987.

[PM90] P. Perona and J. Malik. Scale Space and Edge Detection Using Anisotropic Diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:629–639, 1990.

[Pol84] D. Pollard. *Convergence of Stochastic Processes*. Springer, 1984.

[Pol06] J. Polzehl. *aws: Adaptive Weights Smoothing*, 2006. R package version 1.3-2.

[PS00] J. Polzehl and V.G. Spokoiny. Adaptive Weights Smoothing with Applications to Image Restoration. *Journal of the Royal Statistical Society, Series B*, 62:335–354, 2000.

[ROF92] L.I. Rudin, S. Osher, and E. Fatemi. Nonlinear Total Variation Based Noise Removal Algorithms. *Physica D*, 60:259–268, 1992.

[Rud87] L.I. Rudin. *Images, Numerical Analysis of Singularities and Shock Filters*. PhD thesis, California Institute of Technology, 1987.

[Saa03] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, 2nd edition, 2003.

[Sch93] H.R. Schwarz. *Numerische Mathematik*. B.G.Teubner, 1993.

[Sir54] T. Sirao. On the Uniform Continuity of Wiener Process. *Journal of the Mathematical Society of Japan*, 6:332–335, 1954.

[Tea05] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, 2005. ISBN 3-900051-07-0.

[TW98] A. Tveito and R. Winther. *Introduction to Partial Differential Equations*. Springer, 1998.

[Wah90] G. Wahba. Spline Models for Observatory Data. *CBMS-NSF Regional Conference Series in Applied Mathematics, SIAM*, 59, 1990.

[Wat64] G.S. Watson. Smooth Regression Analysis. *Sankhya, Series A*, 26:101–116, 1964.

[WBSS04] Z. Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing*, 13:600–612, 2004.

[Wei98] J. Weickert. *Anisotropic Diffusion in Image Processing*. B.G.Teubner, 1998.

[WJ95] M.P. Wand and M.C. Jones. *Kernel Smoothing*. Monographs on Statistics and Applied Probability 60. Chapman and Hall, 1995.

[You71] D.M. Young. *Iterative Solution of Large Linear Systems*. Academic Press, 1971.